Scalable Optimization Methods for Machine Learning: Structures, Properties and Applications

A DISSERTATION

SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL

OF THE UNIVERSITY OF MINNESOTA

BY

Shaozhe Tao

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Shuzhong Zhang, Daniel Boley

May, 2018

© Shaozhe Tao 2018
ALL RIGHTS RESERVED

Acknowledgements

First and foremost, I would like to express my sincere gratitude and appreciation to my PhD advisors Prof. Shuzhong Zhang and Prof. Daniel Boley for their invaluable support and guidance. Their broad knowledge, dedication to research and extraordinary advice inspire me throughout my graduate career. I am grateful to them for providing me with tremendous help to explore many research endeavors.

I would like to thank Prof. Qie He, Prof. Zizhuo Wang, Prof. Yousef Saad for serving on my doctoral thesis committee, and Prof. Arindam Banerjee for serving on the committee of my thesis proposal.

I would like to thank Yifan Sun, Xiang Gao, Bo Jiang for collaborations and useful discussions on various research problems. I also would like to thank my friends and colleagues in graduate school: Xiang Gao, Xiaobo Li, Jeff Moulton, Junyu Zhang, Bo Jiang, Shiqian Ma, Yangyang Xu, Guanglin Xu, Tatiana Lenskaia, Sheng Chen, Qilong Gu, Igor Melnyk, Huahua Wang, Farideh Fazayeli, Nicholas Johnson, Konstantina Christakopoulou, Hamidreza Badri, Xiao Chen, Junfeng Zhu, Guiyun Feng, Ruipeng Li, Yuanzhe Xi, Ruoyu Sun, Kai Yu, and many others. I enjoy our discussions on research, technology, culture, and many topics in life. Thank all of you for enriching my life and helping me in many ways.

Most importantly, I thank my parents for their unconditional love. None of this would have been possible without their support.

Dedication

To my parents.

Abstract

Many problems in machine learning can be formulated using optimization models with constraints that are well structured. Driven in part by such applications, the need to solve very large scale optimization models is pushing the performance limits on traditional state-of-art methods. In this thesis, we conduct a systematic study on scalable optimization methods. Our investigations mainly cover three aspects: the role of special structures in the models, convergence properties of algorithms, and applications in machine learning.

First, we study popular scalable methods on sparse structured models, including alternating direction method of multipliers, coordinate descent method, proximal gradient method and accelerated proximal gradient method. In contrast to many global convergence results in the literature, we are particularly interested in the local convergence behavior. We establish the local bounds on the LASSO model, showing their eventual local linear convergence. We show that all of the methods can be treated as some eigenvalue problems, and therefore a spectral analysis becomes applicable. We also observe that when initiated far from the solution, the spectral analysis implies that one possibly get a sequence of iterates that appears to stagnate, but is actually taking small constant steps toward the solution. Moreover, we illustrate how the unaccelerated proximal gradient method can sometimes be faster when the iterates get close enough to the solution, as compared to the accelerated proximal gradient method. A comprehensive comparison of all methods is presented.

Next we move on to group sparse structured model. We develop an inverse covariance estimator that can regularize for overlapping group sparsity, and provide better estimates, especially when the dimension size is much larger than the number of samples. Furthermore, we extend the estimator into a general setting that covers any convex differentiable

functions with conic constraints. The general estimator can exploit the domain structure to reduce the computation cost. The designed Frank-Wolfe method can leverage the decomposition within group structure, hence speeding up computation. Simulations and applications using real data justify both stability and scalability of this estimator, as the results show noticeable improvement.

Finally, we explore a certain low-rank structure in tensor. We construct the connection between the low-rank property in tensor and the group sparsity in its factor matrices. This provides a way to find a low-rank tensor decomposition via a regularized multiconvex optimization. In our approach, no prior knowledge of tensor rank is assumed. We propose to apply block coordinate descent method, since each block update can be implemented efficiently. Consequently, we show that our approach can be used to solve the tensor low-rank completion problem as well.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

There have been spectacular advances in machine learning over the past decades, ranging from WWW-scale social network to genome-scale biological network, from speech and image recognition to self-driving cars, from recommender systems to business analytics. In the new era of big data, machine learning has been playing an ever-increasingly important role in various tasks among not only internet companies but also traditional industries, such as finance, retail, manufacturing, health care, etc.

Essentially, the principle of machine learning is to learn a model through data with algorithms. The model can be formulated in different ways for different applications. Many machine learning problems, from classification and prediction to variable selection can be reduced to optimization problems, or a sequence of optimization problems. Moreover, with a large number of parameters to learn in the models, traditional statistics-based models usually have a tendency to overfit the data. To avoid the overfitting issue, an effective approach is to impose regularization on the variables to reduce the "size" of the search space, thereby reducing the "statistical complexity" of the problem. This involves the result satisfying certain structural conditions, and leads to regularized optimization problems, which will be discussed in Section 1.1.

With the increasing growth of the data and model complexity, the need to solve optimization models arising from machine learning has been pushing the performance limits on the state of art optimization methods. Traditional optimization methods typically require operations such as inverting a dense matrix or computing the second-order derivative. Such operations better be avoided in the new context, simply because they are too expensive to perform. Scalable methods are therefore in high demand and have been received much attention in recent years. How to exploit the structure of models in question with scalable solution methods is the main motivation of this thesis. A more detailed introduction can be found in Section 1.3.

1.1 Optimization Models in Machine Learning

Many machine learning and statistical problems have been shown to fit in the following general form consisting of the loss function f with structural term, either in objective r or constraint $\mathcal{D}$:

$$\min_{x\in\mathcal{D}}\{F(x) = f(x) + r(x)\} \tag{1.1.1}$$

where f is smooth and its gradient is Lipschitz continuous; r is possibly non-smooth function, and $\mathcal{D}$ is a closed set; r and $\mathcal{D}$ are convex; f is convex or multi-convex. Concrete examples will be discussed in Chapter 2.

The class of problems (1.1.1) covers many popular and important problems encountered in machine learning. Function r and constraint $\mathcal{D}$ in (1.1.1) are considered as regularized terms for purpose of structure in the model. One can always set $r = 0$ or $\mathcal{D} = \mathbb{R}^d$ to remove these terms. However, they are important in successfully learning a good model. In machine learning, it is highly possible to train a model very well on the given data but perform poorly on unseen test data, which refers to the overfitting issue. Moreover, many applications of interest arise in situations where samples have high

dimensionality but are expensive to obtain. Hence the task is to learn a huge number of parameters from relatively small amount of data. This can lead to overfitting and statistically unstable result. The main approach is to restrict the parameter space use a structural constraint, such as sparsity, low-rank, linear equations, conic constraint, or a linear combination of a small number of given primitive group structure. And therefore, the result is constructing an optimization problem consisting one part penalizing violations of the training data (function f in (1.1.1)) and one part penalizing violations of the structural assumptions, or a convex relaxation of the structural assumptions (function r and constraint $\mathcal{D}$ in (1.1.1)). Examples include sparse signal recovery [26], image restoration and denoising [9], sparse inverse covariance selection [45], trace norm regularized least squares minimization [69] and so on. We present structural regularizations of our interest in detail in Section 1.2.

In this thesis, we conduct research on the scalable methods for solving optimization models. We focus one three parts: (1) the structures in models; (2) convergence properties of algorithms; and (3) the applications in machine learning. In the rest of this introduction chapter, we shall present a literature review of scalable algorithms, and our contributions, as well as the organization of this thesis.

1.2 Review of Structural Regularization

We introduce the problem to be studied in this thesis:

$$\min_{x\in\mathcal{D}}\{F(x) = f(x) + r(x)\}$$

We assume that such optimization model is endowed with special structures in r and $\mathcal{D}$, which are quite popular in machine learning for different purposes. Below we present special structures of interest.

Unless specified, high order tensors, matrices, vectors and scalars are denoted respectively by calligraphic letters, e.g. $\mathcal{Z}$, capital letters, e.g. Z, boldface lowercase letters, e.g. $\mathbf{z}$, and non-bold lowercase letters, e.g. z.

1.2.1 Sparsity and Norm Regularization

ℓ_1 norm is widely used for sparse models, especially for the application of sparse regression, signal processing, image denoising and social network. Essentially, the original idea is to minimize the number of non-zero elements among all, known as ℓ_0 norm $\|\mathbf{x}\|_0 = \#\{i : x_i \neq 0\}$, the cardinality of number of non-zeros. However, ℓ_0 norm is non-convex and computationally difficult. The optimization problem with ℓ_0 norm in either objective or constraint is NP-hard, and not efficient in computing for large scale data [19, 40, 123]. As a remedy, a popular way is to relax the non-convex regularization into a convex one. Candes and Tao [19] proposed to use ℓ_1 norm to minimize non-zero elements, which is defined as the sum of absolute value, i.e. $\|\mathbf{x}\|_1 = \sum_i |x_i|$. ℓ_1 norm has been proved to be the convex envelope of ℓ_0 norm and is shown to be successful in many problems in both theory and applications [19, 21, 34, 40]. The computation of ℓ_1 is much easier due to its convexity. In later chapters, we will consider it with other loss functions and can be solved efficiently by the so-called shrinkage-thresholding operator.

1.2.2 Structured Sparsity and Latent Group Norm

A natural extension for sparsity is group sparsity. In fact the sparsity pattern in real world doesn't necessarily mean for each element, but for a chunk of data or more complicated special structures. For example, in sparse regression, variables can be categorized into groups and we consider the situation where responses depends on group (or block) of coefficients, not each element independently. The group structure was early explored for applications such as sparse approximation and multi-task learning [4, 124]. Then it

receives a lot of success in more problems [10, 46, 67, 128].

In particular, Yuan and Lin [130] proposed one particular group norm, known $\ell_{1,2}$ norm regularization term. It shows the effect of group norm is better than ℓ_1 regularization in sparse regression in the formulation of group LASSO. Formally, it is defined as

$$\forall \mathbf{x} \in \mathbb{R}^p, \|\mathbf{x}\|_{\ell_{1,2}} = \sum_{g \in \mathcal{G}} w_g \|\mathbf{x}_g\|_2$$

where $\mathcal{G}$ is denoted as all groups, w_g is the positive weight for group $g \in \mathcal{G}$. From the equation, we see $\ell_{1,2}$ norm is the ℓ_1 norm of each group's ℓ_2 norm. Since ℓ_1 norm introduced before has the effect of leading sparsity, $\ell_{1,2}$ can lead sparsity at group level.

Later on, Obozinski, etc [90] developed another type of group norm, known as latent group norm. The motivation is that either ℓ_1 or $\ell_{1,2}$ does not encode any prior knowledge about the structure that one may hope to select together. Given predefined overlapping groups of variables, latent group norm decomposes into a latent variable with each supported by one group. A big advantage of latent factors is that it is capable of accurately recovering the group structure when the groups are overlapping. The definition is as follows:

$$\|\mathbf{x}\|_G = \min \left\{ \sum_{g \in \mathcal{G}} w_g \|\mathbf{v}^g\|_2 : \sum_{g \in \mathcal{G}} \mathbf{v}^g = \mathbf{x} \right\}$$

where w_g is the positive weight for $g \in \mathcal{G}$ and $\mathbf{v}^g$ is considered as latent factor. The sum of all $\mathbf{v}^g$ composing the vector $\mathbf{x}$. Since such decomposition can be done in different ways, a minimization is required to ensure its uniqueness. As a result, the regularization $\|x\|_G$ enforces sparsity for latent factors at group level.

1.2.3 Low-Rank Matrix and Nuclear Norm

Low-rank matrix are pervasive in a broad range of disciples, such as collaborative filtering, multi-task learning, control theory and so on. The direct formulation of minimizing over

a matrix rank is known to be a non-convex and NP-hard problem. A popular heuristic is to replace a rank function with nuclear norm.

The nuclear norm is defined as the sum of singular values, and can be considered as a natural extension for ℓ_1 norm. In context of vectors, ℓ_0 norm is used for low cardinality and ℓ_1 norm is the convex envelope for ℓ_0 norm. As for nuclear norm, it can be understood as the ℓ_1 norm of all singular values of the matrix while rank function is the ℓ_0 norm of all singular values of that matrix. Consequentially, nuclear norm is the convex envelope for rank function. Mathematically, it is defined as

$$\|M\|_* = \sum_{i=1}^{\min\{m,n\}} \sigma_i(M)$$

where $\sigma_i(M)$ is the singular values of M, i.e. the square root of eigenvalues of M^*M. The minimization of nuclear norm is much easier than rank function due to its convexity. Nevertheless, no matter if nuclear norm regularizer is in objective or constraint, the typical way to solve nuclear norm may require a singular value decomposition, which sometimes is still considered expensive if the dimension of matrix is high.

1.2.4 Positive Semdefinite and General Conic Constraint

Conic constraint receives an explosion in interest due to its wide applications in engineering and finance. A set $\mathcal{C}$ is a cone if for every $x \in \mathcal{C}$ and $\theta \geq 0$, we have $\theta x \in \mathcal{C}$. A set $\mathcal{C}$ is a convex cone if for any $x_1, x_2 \in \mathcal{C}$ and $\theta_1, \theta_2 \geq 0$, we have $\theta_1 x_1 + \theta_2 x_2 \in \mathcal{C}$. A set $\mathcal{C}$ is a proper cone if $\mathcal{C}$ is convex, closed, solid (non-empty interior) and pointed (if $x \in \mathcal{C}$ and $-x \in \mathcal{C}$ then $x = 0$).

Most of convex optimization problems be categorized as conic programming, in which minimizing a linear function over a cone. The cone can cover many important examples. We list a few examples we encounter in this thesis.

- Norm cone $\mathcal{C} = \{(x, y) \in \mathbb{R}^{p-1} \times \mathbb{R} \mid \|x\| < y\}$
- Positive semidefinite cone $\mathcal{C} = \{\mathbf{vec}(X) | X \in \mathbb{S}_+^p\}$
- First orthant $\mathcal{C} = \mathbb{R}_+^p$

Conic constraint is especially common for matrix variable as many applications requires the positive semidefiniteness. The standard way to solve conic optimization is to use interior point method, which involves inversing a dimensional related matrix. However, this computation bottleneck restricts people from large dimensions of data. A lot of research has been done on how to avoid direct inverse for different specific problems. In this thesis, we will present a unified framework on how to decompose a large cone into a linear combination of small cones, hence each iteration would be much more efficient.

1.2.5 Tensor and Low-Rank Tensor

Tensor can be considered as an extension from vector and matrix. It is defined as a multi-dimensional array, which naturally presents the structure of real data, such as signal processing, computer vision, data mining, graphical models, and so on.

With the dimension increasing, tensor has different properties from vector and matrix. There are several different ways to define tensor rank. The most popular ones are CP rank (short for CANDECOMP/PARAFAC rank) and Tucker rank. The CP rank of a tensor is defined as the smallest number of rank-one tensors that can generate that original tensor. The rank-one tensor necessarily means it can be written as the outer product of vectors. The Tucker rank is defined based on dimension of core tensor form Tucker decomposition (discussed detailed later). Both CP rank and Tucker rank play an important role in tensor applications. In this thesis, we would like to explore the low CP rank and Tucker rank decomposition and completion.

1.3 Review of Scalable Methods

The scalable optimization methods have attracted significant amount of attractions these years in machine learning. The high dimensionality and/or large number of samples can make each iteration of optimization methods expensive. Hence traditional solvers such as `CVX` [54] or `MOSAK` [3] for solving (1.1.1) no longer satisfy the needs. This is because standard optimization method, such as interior point method, involves inverting a matrix or computing the second-order gradient in each iteration. Such operations are considered expensive and should be avoided.

Fortunately, the big data encountered in applications of science and engineering usually possess or assume special structures such as sparsity or low-rank. In recent years, many scalable algorithms are developed and revisited to take advantage of the special structures of the problems. Among them, the most remarkable algorithms include alternating direction method of multipliers, (accelerated) proximal gradient method, coordinate descent method and Frank-Wolfe algorithm. For each algorithm, we review the standard iterations and demonstrate the theory of convergence property.

1.3.1 Proximal Gradient Method

Proximal gradient method (PG) can be interpreted as one type of fixed point iterations. It dates back to the resolvent of maximal monotone operator in 1970s [100, 101]. It is designed for solving the problem of form (1.1.1) with special structures. The iterate reads as

Algorithm 1 One step of proximal algorithm for (1.1.1)

Input: $x^{[k]}$: k-th iteration; α step size;

1: $x^{[k+1]} = prox_{\alpha r}(x^{[k]} - \alpha \nabla f(x^{[k]}))$

Output: $x^{[k+1]}$.

The proximal operator of a function r, $prox_r : \mathbb{R}^n \to \mathbb{R}^n$, is defined as

$$prox_{\alpha r}(x) = \underset{y}{\operatorname{argmin}} \; r(y) + \frac{1}{2\alpha}\|x - y\|_2^2. \tag{1.3.2}$$

Note that this *prox* operator is well defined since the function minimized on the right hand is strongly convex and thus leads to a unique minimizer. The equivalent explicit expression of PG for (1) is

$$x^{[k+1]} = \underset{x}{\operatorname{argmin}} \; r(x) + \frac{1}{2\alpha}\|x - (x^{[k]} - \alpha\nabla f(x^{[k]}))\|_2^2.$$

We also remark here that this is a broad method. In (1.1.1), if $r = 0$, then PG reduces to

$$x^{[k+1]} = x^{[k]} - \alpha\nabla f(x^{[k]}),$$

exactly the gradient descent method for minimizing f. If $f = 0$, then PG reduces to

$$x^{[k+1]} = prox_{\alpha r}(x^{[k]}) = \underset{x}{\operatorname{argmin}} \; r(x) + \frac{1}{2\alpha}\|x - x^{[k]}\|_2^2,$$

the so-called proximal point method for minimizing r [102, 103] . For general convex function f, the convergence of PG is established to converge with a sublinear rate $O(1/k)$ when the step size $\alpha \in (0, 2/L]$, where L is the Lipschitz constant of ∇f. Linear convergence rate can be established if f is strongly convex. [95].

1.3.2 Accelerated Proximal Gradient Method

Accelerated proximal gradient method (APG) is also intended to solve the problem of form (1.1.1) with $\mathcal{D} = \mathbb{R}^p$. It is a variant of the proximal gradient method of version 1. For each iteration, instead of only use one previous iterate, APG combines two previous iterate in an extrapolation way. The result is an accelerated convergence rate of $O(1/k^2)$

with step size $\alpha \in (0, 1/L]$ [9]. The idea of acceleration was first developed by Nesterov in [89] for convex and differential function f in the objective function only. [9] then extend similar idea for regularized optimization with additional non-differentiable function r in objective function. [120] and reference therein provide a unified study on (2) with other detailed convergence result under various conditions and different parameter choice. The accelerated method reads as follows.

Algorithm 2 One step of accelerated proximal algorithm for (1.1.1)

Input: $x^{[k-1]}, x^{[k-2]}, t^{[k-1]}, t^{[k]}$.

1: $y^{[k]} = x^{[k-1]} + \frac{t^{[k-1]}-1}{t^{[k]}}(x^{[k-1]} - x^{[k-2]})$.

2: $x^{[k]} = prox_{\alpha r}(y^{[k]} - \alpha \nabla f(y^{[k]}))$.

3: $t^{[k+1]} = \frac{1+\sqrt{1+4t^{[k]2}}}{2}$.

Output: $x^{[k-1]}, x^{[k]}, t^{[k]}, t^{[k+1]}$.

Note that above t used for stepsize is a determined sequence starting from $t^{[0]} = 1$. It is not related with the value of each iterate. It is such stepsize update (Step 3 of Algorithm 2) that leads to an accelerated rate of convergence.

1.3.3 Alternating Direction Method of Multiplier

Alternating Direction Method of Multipliers (ADMM) is closely related to the Douglas-Rachford operator splitting method, which has been extensively studied for finding the zeros of the sum of monotone operators [38, 43, 52, 100, 101]. It was proposed in 1950s but revisited recently as it was found to be very efficient for solving problems of the form (1.1.1). The canonical two-block model solved by ADMM:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) + r(\mathbf{z}) \\ \text{s.t.} \quad & A\mathbf{x} + B\mathbf{z} = \mathbf{b} \end{aligned} \tag{1.3.3}$$

with variables $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^p$, where $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times p}$. The associated augmented Lagrangian function of (1.3.3) is

$$\mathcal{L}_\rho = f(\mathbf{x}) + r(\mathbf{z}) + \mathbf{y}^{\mathrm{T}}(A\mathbf{x} + B\mathbf{z} - \mathbf{b}) + \rho/2\|A\mathbf{x} + B\mathbf{z} - \mathbf{b}\|_2^2$$

where ρ is a penalty parameter, $\mathbf{y}$ is the dual variable. Let $\mathbf{u} = \mathbf{y}/\rho$, then ADMM iterate is shown in Algorithm 3.

Algorithm 3 One step of ADMM for (1.3.3)

Input: $\mathbf{x}^{[k]}, \mathbf{z}^{[k]}, \mathbf{u}^{[k]}$.
1: $\mathbf{x}^{[k+1]} = \operatorname{argmin}_{\mathbf{x}} \mathcal{L}_\rho(\mathbf{x}, \mathbf{z}^{[k]}, \mathbf{u}^{[k]})$.
2: $\mathbf{z}^{[k+1]} = \operatorname{argmin}_{\mathbf{z}} \mathcal{L}_\rho(\mathbf{x}^{[k+1]}, \mathbf{z}, \mathbf{u}^{[k]})$.
3: $\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + 1/\rho \nabla_{\mathbf{u}} \mathcal{L}_\rho(\mathbf{x}^{[k+1]}, \mathbf{z}^{[k+1]}, \mathbf{u})$.
Output: $\mathbf{x}^{[k+1]}, \mathbf{z}^{[k+1]}, \mathbf{u}^{[k+1]}$.

Consider the regularized optimization problem of form (1.1.1) with $\mathcal{D} = \mathbb{R}^p$, ADMM is constructed to split the primal $\mathbf{x}$ variable into two separate variables such that the minimum with respect to each individual variable can be easily computed, and then imposing an equality constraint between the two variables. And the Algorithm 3 reduces to Algorithm 4.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) + r(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{x} = \mathbf{z}. \end{aligned} \tag{1.3.4}$$

Algorithm 4 One step of ADMM for (1.3.4)

Input: $\mathbf{x}^{[k]}, \mathbf{z}^{[k]}, \mathbf{u}^{[k]}$.
1: $\mathbf{x}^{[k+1]} = prox_{f/\rho}(\mathbf{z}^{[k]} - \mathbf{u}^{[k]})$.
2: $\mathbf{z}^{[k+1]} = prox_{r/\rho}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]})$.
3: $\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + \mathbf{x}^{[k+1]} - \mathbf{z}^{[k+1]}$.
Output: $\mathbf{x}^{[k+1]}, \mathbf{z}^{[k+1]}, \mathbf{u}^{[k+1]}$.

Consider for problem of form (1.1.1) with $r = 0$, ADMM can be constructed into split the objective function and constraint separately. $\mathcal{I}$ is the indicator function and the

resulting block update is projection over $\mathcal{D}$, shown in Algorithm 5.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) + \mathcal{I}_{\mathcal{D}}(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{x} = \mathbf{z}. \end{aligned} \tag{1.3.5}$$

Algorithm 5 One step of ADMM for (1.3.5)

Input: $\mathbf{x}^{[k]}, \mathbf{z}^{[k]}, \mathbf{u}^{[k]}$.
1: $\mathbf{x}^{[k+1]} = prox_{f/\rho}(\mathbf{z}^{[k]} - \mathbf{u}^{[k]})$.
2: $\mathbf{z}^{[k+1]} = proj_{\mathcal{D}}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]})$.
3: $\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + \mathbf{x}^{[k+1]} - \mathbf{z}^{[k+1]}$.
Output: $\mathbf{x}^{[k+1]}, \mathbf{z}^{[k+1]}, \mathbf{u}^{[k+1]}$.

The benefit of ADMM to separate the constraints and several functions into blocks and each block can be updated in closed form effectively. The simplicity for each block update can be used in diverse applications such as imaging processing and restoration [9], matrix completion [25], consensus computing [41], sparse signal recovery [127] and many others. The two-block can be extended to multi-block for more general applications.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + \cdots + f_N(\mathbf{x}_N) \\ \text{s.t.} \quad & A_1\mathbf{x}_1 + A_2\mathbf{x}_2 + \cdots + A_N\mathbf{x}_N = \mathbf{b} \\ & \mathbf{x}_i \in \mathcal{X}_i \end{aligned} \tag{1.3.6}$$

The standard ADMM update for problem (1.3.6) is then have N primal blocks update for $\mathbf{x}$ plus one more dual block update.

Due to the wide applications of ADMM, its convergence is a hot topic in optimization community these years. The rate of convergence can be established with various conditions. As for two-block ADMM (1.3.3), under some mild conditions such as the non-emptiness of the solution set, Algorithm 3 was proven to be globally sublinear convergent at the rate of $O(1/k)$ for problem (1.3.3) [59]. A locally linear convergence result is shown for

quadratic programming in [11, 60]. If the functions f in objective are strongly convex, A in (1.3.3) is full row rank then global linear convergence is established [33]. Many various conditions may lead to different ADMM convergence rate and we refer readers to [12] for more details. The convergence of multi-block problem in (1.3.6) is different from two-block one. Direct extension of ADMM for multi-block convex optimization is not necessarily convergent [24]. But with several conditions, globally (linear) convergence can still be satisfied [62, 79].

1.3.4 Coordinate Descent Method

Another scalable method that gains great popularity is coordinate descent method. This method is reviewed and developed in recent years due to its competitive performance for solving regularized convex problems. The basic framework of coordinate descent is to fix most components of the variable at their current iterations and minimize over the remaining components. As one could expect, each subproblem has much lower dimension than the main problem, and hence can be solved much more easily [125].

Specifically, for solving (1.1.1), one can take the variable vector $\mathbf{x} \in \mathcal{D}$ as $(x_1, x_2, \cdots, x_p)$. At each step, one minimizes a single component i_k of $F(\mathbf{x})$ at the current iteration, as shown below

$$x_{i_k}^{[k+1]} \in \operatorname{argmin} F_{x_{i_k} \in \mathcal{D}}(x_{i_1}^{[k]}, x_{i_2}^{[k]}, \cdots, x_{i_{k-1}}^{[k]}, x_{i_k}, x_{i_{k+1}}^{[k]}, \cdots, x_{i_p}^{[k]}). \tag{1.3.7}$$

The order of picking components is another topic of great interest in the literature. The order can either be cyclic or randomized. In this thesis, we only consider the cyclic coordinate descent. This is the most obvious and simple way. All of the coordinates will be updated sequentially. However, very little is known about the convergence behavior of cyclic coordinate descent in the literature. [121] proved the global linear convergence

under several error bound conditions and [107] showed the global $O(1/k)$ convergence rate for a more general case. Regarding the convergence of other various types of coordinate descent method, we refer readers to [125] for a very detailed survey on this topic.

1.3.5 Frank-Wolfe Method

The Frank-Wolfe method (aka. conditional gradient method) was first proposed for solving quadratic programming in [44] in 1956. It is popular in large scale optimization due to its high efficiency in constrained optimization of the form (1.1.1) with $r = 0$, i.e. $\min_x\{f(x) : x \in \mathcal{D}\}$. The iteration update reads as follows.

Algorithm 6 One Pass of Frank-Wolfe algorithm

Input: $x^{[k]} \in \mathcal{D}$: k-th iteration. $\eta := \frac{2}{k+2}$ step size.

1: Compute forward step : $s = \arg\min_{s\in\mathcal{D}} \langle s, \nabla f(x^{[k]})\rangle$.

2: Update primal variable : $x^{[k+1]} = (1 - \eta^{[k]})x^{[k]} + \eta^{[k]}s$.

Output: $x^{[k+1]}$.

The convergence of the Frank-Wolfe method is at the rate of $f(x^{[k]}) - f(x^*) \leq O(1/k)$ [35, 44] with x^* being the optimal solution. The forward step also refers to Linear Minimization Oracle (LMO) in some literature. The iterates is a convex combination of previous vertices or "atoms". Usually, the number of such active atoms in iterates is small due to the sparsity of the model.

There are many variants for Frank-Wolfe method. For example, the stepsize η can be tuned through line search. The forward step (LMO) can be obtained through inexact minimization. The iterates could be updated by $x^{[k+1]} = \operatorname{argmin}_{x\in\mathcal{D}} \operatorname{conv}(s^{[0]}, \cdots, s^{[t+1]})$ known as fully corrective. The convergence rate could be faster for further assumptions over f. We refer readers to [76] for these results. A typical one is if f is strongly convex and $\mathcal{D}$ is polytope, the iterates will converge linearly.

Frank-Wolfe method has great advantage in computation when the domain $\mathcal{D}$ enjoy special structure. For vector variables, when the domain $\mathcal{D}$ is $\|\cdot\|_1$-ball, then the

corresponding update is $\|\cdot\|_\infty$. If $\mathcal{D}$ is $\|\cdot\|_p$ ball, then the update is $\|\cdot\|_q$ with $1/p + 1/q = 1$. For matrix variables, when the domain is trace norm ball, the update is largest singular value. We will take this advantage when we develop this algorithm in a scalable way later in this thesis.

1.4 Contributions and the Organization

With previous introduction, we now start the discussion of our interests and contributions in exploring the scalable algorithms for optimization with structure in machine learning.

First, we study sparse model and use model LASSO problem for analysis in our work [112, 113]. We review that all of the mentioned algorithms enjoy a global sublinear convergence. In contrast, we are particularly interested in the local convergence of these algorithms. We establish the local bounds of proximal gradient, accelerate proximal gradient, ADMM and coordinate descent on a model LASSO problem. We compare the asymptotic convergence behavior of these methods and show that linear convergence can be reached eventually, but not necessarily from the beginning. As for proximal gradient, accelerated proximal gradient and ADMM, the method is based on a representation of a matrix recurrence. As for coordinate descent, we show its close relation to the Gauss Seidel method applied on the linear system. Hence all of the algorithms can be treated as the eigenvalue problems and spectral analysis would be applied. Note that our analysis doesn't require strong convexity but much more mild conditions such as strict complementarity and unique solution. We show that, the iterations of all of the methods will pass through several stages or "regimes" of different types, some of which consist of taking constant steps, but finally reaching a regime of linear convergence when close enough to the optimal solution.

Apart from establishing local linear convergence theory, our analysis provide a way to study the properties of iterations, especially for the comparison between proximal

gradient and accelerated proximal gradient on the LASSO problem. Though the latter can be considered an accelerated version, we show that as one approaches the solution accelerated proximal gradient can slow down and even become slower than proximal gradient. Moreover, from our spectral analysis, the convergence rates are bounded by the eigenvalues of the corresponding matrix operator. This helps explain why accelerated proximal gradient iterations oscillates towards the end but proximal gradient doesn't. Such new discovered properties makes our analysis complementary to the existing global convergence analysis. A heuristic algorithm is developed to take advantage of these properties.

Next we move on to group sparse model in our work [114, 115]. We investigate the group structure on Sparse Inverse Covariance Estimation. We develop a novel estimator that can both exploit the domain knowledge of structure information and reduce the computation cost. In the estimator, we incorporate the so-called latent group norm and define them as principal submatrices, which can handle the overlapping group structure. Taking advantage of the special structure, we show that Frank-Wolfe method can leveraging the decomposition for scalability. Simulation results show siginifcant improvement in sample complexity when the correct group structure is known. We also apply the estimator to stock closing prices and congressman voting history, with noticeable improvement when group sparsity is exploited.

Furthermore, we show the developed group sparse estimator can be extended into a general setting. The novel model consider any convex differentiable function with any conic constraint. The group norm is extended from ℓ_2 to any valid norm, and the corresponding optimization procedure can utilize the decomposition of group structure. If we can assume any solution is a linear combination of a small set of "chunks", the cost can be reduced to an amount dependent on the size of each chunk times the number of chunks. Moreover, the computation of each chunk is simply computing the dual of the

defined norm, which can save a lot of cost. For example, if the defined matrix norm is nuclear norm, then its dual is its maximal singular value and can be efficiently solved by Lanczos or other Krylov subspace methods. We also present more model problems, such as sparse matrix nearness problem and sparse trace regression problem, for future work.

Finally, we study the low-rank structure in tensor. In our work [49], we investigate tensor low-rank decomposition and completion problem. A well-known state-of-art method to compute the decomposition is known as the alternating least squares algorithm, which first proposed by Carroll and Chang [22]. There are some other traditional methods including derivative-based algorithms (dGN, PMF3) and ASD [71]. It should be noticed that all these methods require the rank of a tensor. However, In our proposed method, we do not assume prior knowledge regarding the rank of tensor to be recovered. The key underlying idea is to take advantage of group sparsity structure. We set up a connection between group structure and low-rank, so that we formulate a regularized multi-convex optimization. The resulting objective consists of two parts. One is the least square term for the traditional tensor decomposition, while the other one is the group sparse regularization term that would lead to a low-rank solution. We apply the Gauss-Seidel type of block coordinate descent and each subproblem has closed form solution by proximal method. We also show that our approach can be used to solve the tensor low-rank completion problem. Simulation results demonstrate that our new method performs well numerically, especially when the tensor to be recovered indeed has a low-rank structure.

The organization of this thesis is described as follows:

- In Chapter 2, we present motivation examples and preliminaries of this thesis.
- In Chapter 3, we use a model LASSO problem to analyze the convergence of proximal gradient (PG) and accelerated proximal gradient (APG) iterations. We use more popular terms ISTA and FISTA to stand for PG and APG respectively,

when solving the LASSO problem. Using a spectral analysis of the associated matrix operators through Sections 3.2 - 3.4, we show that both iterations satisfy local linear convergence bound when close enough to the solution. Moreover, in Section 3.5, our analysis points out FISTA can slow down as it proceeds, and eventually becoming slower than ISTA.

- Chapter 4 is dedicated to ADMM and Coordinate Descent for the LASSO problem. We extend the same technique in the last chapter, showing that linear convergence of ADMM is reached eventually for the LASSO problem in Section 4.2. We also establish a natural relationship between the iterations of CD and the Gauss-Seidel method so that linear convergence is guaranteed towards the end of CD in Section 4.3. Finally, through numerical examples in Section 4.4, we compare the convergence behavior of all scalable algorithms.

- In Chapter 5, we discuss the inverse covariance estimation with group structure. Section 5.1 reviews the standard sparse inverse covariance problem, existing popular methods known as Graphical LASSO, and their issues. Section 5.2 defines a novel group norm for matrix, given prior knowledge of structural information. Our estimator is described in Section 5.3. Section 5.4 discusses an efficient way of applying the Frank-Wolfe method for this problem. Simulation results are presented in Section 5.5. We further show the performance of this estimator on real world data in Section 5.6 and Section 5.7.

- Chapter 6 generalizes the estimator in Chapter 5. Section 6.1 talks about a general type of group norm. Then the formal estimator with group structure is demonstrated in Section 6.2. A scalable optimization procedure based on Frank-Wolfe method is shown in Section 6.3. We present more models in Section 6.4.

- In Chapter 7, we provide a novel way to find low-rank tensor decomposition without

any prior knowledge of the recovered tensor rank. We construct the connection between group structure in factor matrix and low-rank tensor in Section 7.2. Then in Section 7.3, tensor decomposition is formulated via a multi-block optimization model and solved by block coordinate update. We also show in Section 7.4 that our approach can be applied to the tensor low-rank completion problems.

- Chapter 8 summarizes the contribution and main results of this thesis.

Chapter 2

Preliminaries: Structured Models

In this chapter, we present several examples of form (1.1.1) with preliminary knowledge. The matrix p-norm is the norm induced by the corresponding vector norm: $\|M\|_p = \max_{\|\mathbf{v}\|_p=1} \|M\mathbf{v}\|_p$, with $p = 1, 2$ or ∞. We use σ to denote the singular values of a matrix M. $\mathbb{S}^p$ denotes symmetric matrix. $\mathbb{S}^p_+$ and $\mathbb{S}^p_{++}$ denote positive semidefinite matrix and positive definite matrix respectively.

2.1 The LASSO Model

Over recent decades, ℓ_1-norm regularized least squares could be one of the most influential models in many areas such as compressive sensing [26], statistics [39], sparse coding [55], geophysics [116] and so on. In compressive sensing, we seek to recover a solution $\mathbf{x}$ to an underdetermined $n \times p$ system of linear equations $A\mathbf{x} = \mathbf{b}$ with $p \gg n$. This linear system either has no solution (if A is rank-deficient) or has many solutions. A common approach is to find the solution with minimum ℓ_2-norm. However, it is often desired to find a solution $\mathbf{x}$ with as few non-zero entries as possible, as a way to find the fewest columns of A needed to obtain a good approximation of the target vector $\mathbf{b}$. A straightforward

way to obtain a sparse solution $\mathbf{x}$ is to use ℓ_0 norm.

$$\min_{\mathbf{x}\in\mathbb{R}^p} \|\mathbf{x}\|_0 \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{2.1.1}$$

Problem (2.1.1) is not convex and by now it is well known to be approximated by ℓ_1 regularization as below.

$$\min_{\mathbf{x}\in\mathbb{R}^p} \|\mathbf{x}\|_1 \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{2.1.2}$$

Its Lagrangian form is known as the "Least Absolute Shrinkage and Selection Operator" (LASSO) problem.

$$\min_{\mathbf{x}\in\mathbb{R}^p} F(\mathbf{x}) = \tfrac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{2.1.3}$$

where $A \in \mathbb{R}^{n\times p}$ is a given matrix, $\mathbf{b}$ is a given vector and λ is a positive tuning parameter. The objective is convex consisting of a least square function plus a non-smooth ℓ_1 norm, in the form of (1.1.1) with no constraint. Since many problems can be set up as attempting to find sparse approximation, LASSO is widely studied in [15, 16, 26, 85, 93, 117] with applications in [5, 46, 47, 55, 129, 132], to name a few. Throughout this thesis, we will use LASSO as a model problem to study a wide class of scalable algorithms including (accelerating) proximal gradient method, alternating direction method of multipliers, coordinate descent method.

Now we demonstrate preliminaries of LASSO problem including the optimality conditions, uniqueness and strict complementarity that would be referred to occasionally in this thesis.

2.1.1 The Optimality Condition

The first order KKT optimality conditions for the LASSO problem (2.1.3) are

$$A^{\mathrm{T}}(\mathbf{b} - A\mathbf{x}) = \lambda\boldsymbol{\nu} \tag{2.1.4}$$

where at optimality each component of $\boldsymbol{\nu}$ satisfies

$$\left\{\begin{array}{ll} \nu_i = \text{sign}\,(x_i) & \text{if } x_i \neq 0 \\ -1 \leq \nu_i \leq +1 & \text{if } x_i = 0 \end{array}\right\} \qquad \text{for} \quad i = 1, 2, \cdots . \tag{2.1.5}$$

Here the "Sign" function is defined as

$$\text{sign}\,(x) = \begin{cases} +1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0. \end{cases}$$

2.1.2 Uniqueness and Strict Complementarity

There are various sufficient and necessary conditions for the uniqueness of the LASSO problem or its variants. For example, [16, 47, 93] show different sufficient conditions and [118] studies the necessary conditions for the LASSO problem. In fact, the problem (2.1.3) needs to have a unique solution in many situations. For example, in compressive sensing and signal recovery, having non-unique solutions results in unreliable recovery given the data. We refer readers to [118, 132] and references therein for a discussion of the uniqueness of the LASSO problem. Here we will occasionally assume uniqueness of the LASSO solution.

Let $\mathbf{x}^*$ be an optimal solution to (2.1.3) with corresponding residuals $\boldsymbol{\nu}^*$ (2.1.4). We say this solution has the property of *strict complementarity* if for every index i, the i-th components of $\mathbf{x}^*, \boldsymbol{\nu}^*$ satisfy either $x_i^* > 0$ and $\nu_i^* = \text{sign}\,(x_i) = \pm 1$ or else $x_i^* = 0$ and $|\nu_i^*| < 1$. The situation where $x_i^* = 0$ and $|\nu_i^*| = 1$ for the same index i would violate the condition of strict complementarity. We remark that such violations occur only for finitely many values of λ for a given A and $\mathbf{b}$ [39].

2.2 Low-Rank Matrix Recovery Problem

After the success in the Netflix challenge [74], collaborative filtering has shown a big success in business [99]. In the general setting, (i, j) denotes the index for user and item and hence M_{ij} represents the rating that user i gives to item j. Items could be articles in a content recommendation portal, suggested movies in sites like Netflix, items to buy in Amazon. Ratings could be clicks to an item or explicit numbers in a scale. Typically only a small portion of M is known. The goal is to predict the missing entries of M. A common assumption is that only a few factors influence users' ratings. Hence M is believed to have a low-rank structure. Recovering missing values of the low-rank matrix can be formulated as

$$\min_{X \in \mathbb{R}^{m \times n}} \{\mathbf{rank}(X) : X_{ij} = M_{ij}\} \tag{2.2.6}$$

Problem (2.2.6) is NP-hard due to the noncontinuous nature of the rank function. There are two principal ways to approximate a solution.

Nuclear Norm Regularized Formulation

On one hand, as mentioned early, (2.2.6) can be relaxed as a regularized convex optimization problem by replacing the rank function with its convex envelope, the nuclear norm.

$$\min_{X \in \mathbb{R}^{m \times n}} \{\|X\|_* : \mathcal{A}(X) = \mathbf{b}\} \tag{2.2.7}$$

or its equivalent Lagrangian formulation

$$\min_{X \in \mathbb{R}^{m \times n}} 1/2 \|\mathcal{A}(X) - \mathbf{b}\|_2^2 + \mu \|X\|_*. \tag{2.2.8}$$

where $\mathcal{A} : \mathbb{R}^{m \times n} \to \mathbb{R}^p$ is a linear map and μ is a positive scalar, related with M. Other than standard convex solvers by reformulating into semidefinite programming, an efficient way to solve (2.2.7) and (2.2.8) is using the singular value thresholding operator for

matrix [14]. Given threshold τ, each iteration requires a singular value decomposition, i.e. $X = U\Sigma V^*$ with Σ the diagonal matrix of all singular values σ_i. The singular value thresholding operator then shrinkage σ_i by defining as $\mathcal{D}_\tau(\Sigma) = \mathbf{diag}(\{(\sigma_i - \tau)_+\})$. As for convergence, the sublinear rate can be obtained. The paper [119] developed a faster proximal gradient method using the Nesterov acceleration update. Under certain conditions, a linear convergence rate can be established [64]. Since each iteration involves a singular value decomposition, the cost per iteration is relative high for large matrix. An alternative formulation is established in the literature.

Matrix Factorization Formulation

Previous formulation uses nuclear norm to enforce low-rank. In matrix factorization, we assume X has low-rank s and can be written as $X = UV^T$, with $U \in \mathbb{R}^{m\times s}$ and $V \in \mathbb{R}^{n\times s}$. U and V represent user-specific and item-specific latent feature vectors. Clearly the low-rank of X is guaranteed under this setting. Assuming Ω is the set containing existing values (i, j) as before, the formulation [74] is written as

$$\min_{U,V} \tfrac{1}{2}\|\mathcal{P}_\Omega(M - UV^T)\|_F + \tfrac{\alpha}{2}\|U\|_F^2 + \tfrac{\beta}{2}\|V\|_F^2$$

where $\|\cdot\|_F$ denotes Frobenius norm, α and β are weights. $\mathcal{P}_\Omega(M - UV^T)$ is the penalty loss function for projecting over all existing values. Regularizers $\|U\|_F^2$ and $\|V\|_F^2$ are added to ensure stability. The ultimate goal is to find the latent factors U, V, which are used for predicting missing values. And the resulting X can also be obtained.

The formulation (2.2) still in the form of (1.1.1) but the loss function is jointly convex. Therefore finding a global optimum in limited time is intractable in general. Many algorithms are trying to look for a stationary point. There are two major ways to solve this problem. One is (stochastic) gradient descent [50]. The other one is alternating minimization [74]. There has been many discussions between this two methods and other

variants in recent years. Note that latent variables U and V are jointly convex, which means if one variable is fixed, then solving the other one is a convex problem. Hence alternating minimization in this problem can be considered as a special case of two block coordinate descent. And this coordinate descent method can be extended for multi-block update and has wide applications in tensor problems.

2.3 Tensor Factorization and Completion

Both tensor decomposition or completion can be formulated into regularized optimization of the form (1.1.1) (see e.g. [126]).

$$\min_x f(x_1, \cdots, x_N) + \sum_{i=1}^{N} r_i(x_i) \tag{2.3.9}$$

where $x_1, \cdots, x_N$ is the decision variables, f is assumed to be a differentiable and block multiconvex function, that is, f is a convex function of x_i while all the other blocks are fixed, and $r_i, i = 1, \cdots, N$ are convex functions.

To be more specific, $x_1, \cdots, x_N$ can be considered as factor matrices $A_1, A_2, \cdots, A_N$ in tensor. Given tensor $\mathcal{X}$ to be factorized, the resulting problem is to find factor matrices in the following form.

$$\min_{A_1, A_2, \cdots, A_N} \frac{1}{2}\|\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!]\|^2 + \rho\, \mathbf{rank}([\![A_1, A_2, \cdots, A_N]\!]). \tag{2.3.10}$$

Similarly, tensor completion can be formulated into

$$\min_{A_1, A_2, \cdots, A_N} \frac{1}{2}\|\mathcal{P}_\Omega(\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!])\|^2 + \rho\, \mathbf{rank}([\![A_1, A_2, \cdots, A_N]\!]) \tag{2.3.11}$$

where Ω indicates the set of given data and $\mathcal{P}$ is a projection operator.

The general method for solving (2.3.9) is known as block coordinate descent algorithm.

In either Gauss-Seidel type or Jacobi type of update, one minimizes the objective function over one block while fixing remaining blocks. Then each block minimization turns out to be another regularized problem of the form (1.1.1) with convex objective and no constraint. In this thesis, we will explore low-rank structures of tensor and use sophisticated but scalable algorithms to solve them.

2.4 Sparse Inverse Covariance Estimation

The inverse covariance matrix, also known as precision matrix, is of interest to statisticians in biology, finance, machine learning, etc. In graphical models, for p random variables with true covariance matrix $C \in \mathbb{S}^p_{++}$, the sparsity pattern of C^{-1} gives the conditional independence between each pair of variables. However, if $n \ll p$, then the sample covariance matrix $\hat{C}$ is not invertible, and the pseudoinverse $\hat{C}^\dagger$ is inaccurately dense.

Specifically, if C the $p \times p$ positive definite symmetric matrices is the true covariance matrix for normally distributed random variables $x_1, x_2, \ldots, x_p$, then $(C^{-1})_{ij} = 0 \iff x_i \perp\!\!\!\perp x_j \mid x_k, k \neq i, j$, i.e., x_i is independent of x_j conditioned on all other variables x_k. The use of the precision matrix C^{-1} in modeling dates back to [32], who argued that parsimony in the precision matrix better characterizes variable relatedness than that in the covariance matrix C. However, given i.i.d. random samples $\{z_1, z_2, \cdots, z_n\}$ from an unknown distribution of z, the natural estimator of the covariance matrix

$$\hat{C} = \frac{1}{n}\sum_{k=1}^{n}(z_k - \bar{z})(z_k - \bar{z})^T \quad \text{where } \bar{z} = \frac{1}{n}\sum_{k=1}^{n} z_k$$

is singular and unstable for estimation in high dimensions ($n \ll p$). Hence the estimated precision matrix would be even more difficult to obtain.

The most popular approach is ℓ_1 regularized negative log likelihood estimator, also

referring to the graphical LASSO estimator [45], the solution to

$$\begin{array}{ll} \underset{X}{\text{minimize}} & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) + \rho\|X\|_1 \\ \text{s.t.} & X \succeq 0 \end{array} \tag{2.4.12}$$

for some regularization parameter $\rho > 0$. Recall our general form (1.1.1), $f(X) = -\text{logdet}(X) + \text{tr}\,(\hat{C}X)$, $r(X) = \rho\|X\|_1$ and $\mathcal{D} = \{X : X \succeq 0\}$. In fact $f(X)$ is the maximum likelihood function and $r(X)$ is used to induce sparsity.

The main issue of graphical LASSO is the computational complexity of evaluating (2.4.12) or its derivative, especially when the dimensionality p is very large. Most work has been proposed to improve the scalability of algorithms to solve (2.4.12), e.g., first-order methods [29, 108] and block coordinate descent methods [7, 29, 45, 110].

The high dimensional consistency bound of (2.4.12) was first addressed by [106] under certain conditions including the Gaussian distribution and a lower bound on its eigenvalues. The paper [77] then provided the analysis with more general regularization terms and [96] generalized the sample complexity result to random vectors with heavy tailed distributions. There are other variants of (2.4.12). The paper [46] grouped together all edges connected to a given node resulting in a sparse penalty on the nodes instead of edges. The paper [28] considered solving a chordal graph pattern via a second order Newton's method.

Chapter 3

Local Linear Convergence of ISTA and FISTA on the LASSO Model

In Section 2.1, we give a thorough introduction on the LASSO model, with its optimality condition and uniqueness of solution. In this chapter, we use a LASSO model to analyze the convergence behavior of the proximal gradient method and accelerated proximal gradient method. The proximal gradient for LASSO is also known as Iterative Shrinkage Threshold Algorithm (ISTA) and accelerated proximal gradient is known as Fast ISTA (FISTA). Throughout numerical linear algebra analysis, we show that both iterations satisfy local linear convergence rate bound when close enough to the solution. Using the observation that FISTA is an accelerated ISTA process, and a spectral analysis of the associated matrix operators, we show that FISTA's convergence rate can slow down as it proceeds, eventually becoming slower than ISTA. We illustrate the results with some synthetic numerical examples.

This chapter is organized as follows. We first introduce the basic iterations of ISTA and FISTA on the LASSO problem and review their convergence in Section 3.1. Then we show how to represent ISTA as a matrix recurrence in Section 3.2.1 and give some of its

spectral properties in Section 3.2.2, and do the same for FISTA in Sections 3.2.3 & 3.2.4. Section 3.3 gives details about four types of regimes that ISTA and FISTA will encounter in the iterations process based on our spectral analysis. Our first main result is given in Section 3.4, which shows the local linear convergence of ISTA and FISTA on the LASSO problem. In Section 3.5 we compare the behavior in each regime, showing that FISTA can be faster that ISTA through most of the regimes, but asymptotically can be slower as one approaches the optimal solution. Section 3.6 includes two numerical examples run by the standard ISTA and FISTA, to illustrate many of the predicted behaviors.

Throughout this chapter, the matrix p-norm is the norm induced by the corresponding vector norm: $\|M\|_p = \max_{\|\mathbf{v}\|_p=1} \|M\mathbf{v}\|_p$, with $p = 1, 2$ or ∞. We use $\rho(M)$ to denote the spectral radius (largest absolute value of any eigenvalue of a matrix M). For any real symmetric matrix M, the matrix 2-norm is the same as the spectral radius: $\|M\|_2 = \rho(M)$, hence we use those interchangeably for symmetric matrices. We also use a so-called G-norm where G is a non-singular matrix, defined to be $\|\mathbf{v}\|_G = \|G\mathbf{v}\|_\infty$ for any vector $\mathbf{v}$, and $\|M\|_G = \|GAG^{-1}\|_\infty$ for any matrix M.

3.1 ISTA and FISTA Iterations

Proximal gradient method for solving the LASSO problem is known as iterative shrinkage thresholding algorithm (ISTA) because the iteration involves the shrinkage operator. The accelerated proximal gradient method for the LASSO problem is also called as fast ISTA or FISTA. Since ISTA and FISTA are more commonly called when solving the LASSO problem, we may use the term ISTA and FISTA most of the time in the rest of the chapter.

Essentially, ISTA is a gradient descent type step with a penalty to limit the length of each step. Its computation involves only matrix-vector multiplications, which has great advantage over many other convex algorithms by avoiding a matrix factorization [95].

Beck and Teboulle [9] proposed an accelerated ISTA, named as FISTA, in which a specific relaxation parameter is chosen. The idea of acceleration was first developed by Nesterov in [89]. Both algorithms are designed for solving more general problems containing convex differentiable objectives combined with an ℓ_1 regularization term in the following general form:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) + g(\mathbf{x}),$$

where f is a smooth convex function and g is a continuous convex function but possibly nonsmooth. Clearly, the LASSO problem is a special case of the above formulation with $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|^2$, $g(\mathbf{x}) = \|\mathbf{x}\|_1$. The gradient $\nabla f = A^{\mathrm{T}}A\mathbf{x} - A^{\mathrm{T}}\mathbf{b}$ is Lipschitz continuous with constant $L(f) = \rho(A^{\mathrm{T}}A) = \|A^{\mathrm{T}}A\|_2$, i.e., $\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L(f)\|\mathbf{x}_1 - \mathbf{x}_2\|, \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^p$. It has been shown [9] that FISTA provides a convergence rate of $O(1/k^2)$ compared to the rate of $O(1/k)$ by ISTA, while maintaining practically the same per-iteration cost, where k is the iteration number.

In contrast to the global convergence rate, there have been several previous studies of the local convergence of ISTA. Bredies and Lorenz in [13] provided the local linear convergence of the iterative soft-thresholding algorithms in infinite dimensional Hilbert spaces under certain conditions. Recently, Liang et al. [78] showed that a general forward-backward proximal splitting algorithm converges linearly if the function satisfies the so-called partly smooth properties. The local linear convergence of FISTA is not clear in the literature. A Lyapunov analysis on the local convergence FISTA has been recently established in [94].

We establish local bounds on the convergence behavior of ISTA and FISTA on the model LASSO problem. We compare the asymptotic convergence behavior of ISTA with that of FISTA. The latter can be considered an accelerated ISTA, but we show that as one approaches the solution FISTA can slow down and even become slower than ISTA. Extending the same techniques as in [11, 112], we show that linear convergence can be

reached eventually, but not necessarily from the beginning. The method is based on a representation of ISTA and FISTA as a matrix recurrence and a spectral analysis of that matrix recurrence. This yields a model that shows the iterations pass through several phases or "regimes", each treated separately in terms of the spectral model. After passing through several regimes, some of which consist of taking constant steps, each iteration reaches a regime of linear convergence with a convergence rate bounded by the eigenvalues of the corresponding matrix operator. Unlike [13, 78], our analysis is conducted on the finite dimensional Euclidean space so that we can take advantage of the well-established matrix properties. This lets us study the local convergence of not only ISTA but also FISTA, which is not considered in [13, 78]. Apart from the local convergence results, our analysis in terms of regimes can model the behavior of entire iteration process and give a way to compare ISTA and FISTA with each other.

Recall the LASSO problem is

$$\min_{\mathbf{x}\in\mathbb{R}^p} F(\mathbf{x}) = \tfrac{1}{2}\|A\mathbf{x}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{3.1.1}$$

For ease of comparison between ISTA and FISTA, we let $\widehat{\mathbf{x}}$ and $\widetilde{\mathbf{x}}$ denote the iterates of ISTA and FISTA respectively. The basic step of ISTA for the LASSO problem can be reduced to [9, 30]

$$\begin{aligned}
\widehat{\mathbf{x}}^{[k+1]} &= \operatorname*{argmin}_{\widehat{\mathbf{x}}}\{g(\widehat{\mathbf{x}}) + \tfrac{L}{2}\|\widehat{\mathbf{x}} - (\widehat{\mathbf{x}}^{[k]} - \tfrac{1}{L}\nabla f(\widehat{\mathbf{x}}^{[k]}))\|^2\} \\
&= \operatorname*{argmin}_{\widehat{\mathbf{x}}}\{\lambda\|\widehat{\mathbf{x}}\|_1 + \tfrac{L}{2}\|\widehat{\mathbf{x}} - (\widehat{\mathbf{x}}^{[k]} - \tfrac{1}{L}(A^{\mathrm{T}}A\widehat{\mathbf{x}}^{[k]} - A^{\mathrm{T}}\mathbf{b})\|^2\} \\
&= \mathrm{Shr}_{\lambda/L}((I - \tfrac{1}{L}A^{\mathrm{T}}A)\widehat{\mathbf{x}}^{[k]} + \tfrac{1}{L}A^{\mathrm{T}}\mathbf{b}).
\end{aligned}$$

Here the shrinkage operator is defined in terms of a positive threshold ξ:

$$\mathrm{Shr}_{\xi}(s) = \begin{cases} s - \xi & \text{if } s \geq \xi \\ 0 & \text{if } -\xi < s < \xi \\ s + \xi & \text{if } s \leq -\xi, \end{cases} \tag{3.1.2}$$

where this is applied elementwise to vectors. One advantage of ISTA is that the above step can be solved in closed form using only matrix-vector multiplications, leading to the following updates repeated until convergence. Here L is a constant such that $L \geq \|A^{\mathrm{T}}A\|_2$, though for the ISTA analysis it could be as low as $1/2\|A^{\mathrm{T}}A\|_2$. We show one pass through the algorithm, with $\widehat{\mathbf{x}}^{[k]}$ denoting the vector from previous pass and $\widehat{\mathbf{x}}^{[k+1]}$ denoting the new iterate.

Algorithm 7 One pass of Proximal Gradient Method (ISTA)

Input: Start with $\widehat{\mathbf{x}}^{[k]}$.

1: Set $\widehat{\mathbf{x}}^{[k+1]} = \mathrm{Shr}_{1/L}((I - 1/L A^{\mathrm{T}}A)\widehat{\mathbf{x}}^{[k]} + 1/L A^{\mathrm{T}}\mathbf{b})$.

Output: $\widehat{\mathbf{x}}^{[k+1]}$ for next pass.

APG or FISTA differs from ISTA in that the shrinkage operator is employed not on the previous point $\widetilde{\mathbf{x}}^{[k-1]}$ but a different point $\mathbf{y}^{[k]}$, which uses a very specific linear combination of the previous two points $\{\widetilde{\mathbf{x}}^{[k-1]}, \widetilde{\mathbf{x}}^{[k-2]}\}$. The algorithm of FISTA for the LASSO problem is presented below, with the initial $\mathbf{y}^{[1]} = \widetilde{\mathbf{x}}^{[0]} \in \mathbb{R}^p$ and $t^{[0]} = t^{[1]} = 1$.

Algorithm 8 One pass of Accelerated Proximal Gradient Method (FISTA)

Input: Start with $t^{[k]}$, $t^{[k-1]}$, $\widetilde{\mathbf{x}}^{[k-1]}$ and $\widetilde{\mathbf{x}}^{[k-2]}$.

1: Set $\mathbf{y}^{[k]} = \widetilde{\mathbf{x}}^{[k-1]} + \frac{t^{[k-1]}-1}{t^{[k]}}(\widetilde{\mathbf{x}}^{[k-1]} - \widetilde{\mathbf{x}}^{[k-2]})$.

2: Set $\widetilde{\mathbf{x}}^{[k]} = \mathrm{Shr}_{1/L}((I - 1/L A^{\mathrm{T}}A)\mathbf{y}^{[k]} + 1/L A^{\mathrm{T}}\mathbf{b})$.

3: $t^{[k+1]} = \frac{1+\sqrt{1+4t^{[k]\,2}}}{2}$.

Output: $t^{[k+1]}$, $t^{[k]}$, $\widetilde{\mathbf{x}}^{[k]}$, $\widetilde{\mathbf{x}}^{[k-1]}$for next pass.

3.2 Auxiliary Variables with Local Monotonic Behavior

We reform ISTA and FISTA into matrix recurrence formulations by auxiliary variables.

3.2.1 ISTA as Matrix Recurrence

Recall the ISTA iteration for the LASSO problem is

$$\widehat{\mathbf{x}}^{[k+1]} = \mathrm{Shr}_{\lambda/L}((I - 1/L A^{\mathrm{T}}A)\widehat{\mathbf{x}}^{[k]} + 1/L A^{\mathrm{T}}\mathbf{b})$$

where L is the gradient Lipschitz constant for least square term. To reformulate, instead of using variables $\widehat{\mathbf{x}}^{[k]}$, we use two auxiliary variables to carry the iteration. One variable, namely, $\widehat{\mathbf{w}}^{[k]}$ exhibits smooth behavior, with linear convergence locally around a fixed point, and the other variable $\widehat{\mathbf{d}}^{[k]}$ is simply a ternary vector based on the three cases of the shrinkage operator. We let, for all k, the common iterate be

$$\widehat{\mathbf{w}}^{[k]} = (I - 1/L A^{\mathrm{T}}A)\widehat{\mathbf{x}}^{[k]} + 1/L A^{\mathrm{T}}\mathbf{b} \tag{3.2.3}$$

and vector $\widehat{\mathbf{d}}^{[k]}$ is defined elementwise as

$$\widehat{d}_i^{[k]} = \mathrm{sign}\left(\mathrm{Shr}_{\lambda/L}\widehat{w}_i^{[k]}\right)) = \begin{cases} 1 & \text{if } \widehat{w}_i^{[k]} > \lambda/L \\ 0 & \text{if } -\lambda/L \leq \widehat{w}_i^{[k]} \leq \lambda/L \\ -1 & \text{if } \widehat{w}_i^{[k]} < -\lambda/L. \end{cases} \tag{3.2.4}$$

We also define the matrix $\widehat{D}^{[k]} = \mathrm{diag}(\widehat{\mathbf{d}}^{[k]})$. Since $\widehat{D}^{[k]}$ indicates the sign of the iterates, we call it flag matrix in the rest of the thesis. By the updating rule in Alg. 1

and the above two equations, one can obtain the $\widehat{\mathbf{x}}^{[k]}$-update in terms of $\widehat{\mathbf{w}}^{[k]}$ and $\widehat{\mathbf{d}}^{[k]}$

$$\widehat{\mathbf{x}}^{[k+1]} = \mathrm{Shr}_{\lambda/L}(\widehat{\mathbf{w}}^{[k]}) = (\widehat{D}^{[k]})^2\widehat{\mathbf{w}}^{[k]} - \lambda/L\,\widehat{\mathbf{d}}^{[k]}. \tag{3.2.5}$$

Using (3.2.3), (3.2.4) and (3.2.5), the update formula for $\widehat{\mathbf{w}}$ now can be expressed explicitly as follows:

$$\begin{aligned}\widehat{\mathbf{w}}^{[k+1]} &= R^{[k]}\widehat{\mathbf{w}}^{[k]} + \mathbf{h}^{[k]} \\ &= [(I - 1/L\,A^{\mathrm{T}}A)(\widehat{D}^{[k]})^2]\widehat{\mathbf{w}}^{[k]} - (I - 1/L\,A^{\mathrm{T}}A)\lambda/L\,\widehat{\mathbf{d}}^{[k]} + 1/L\,A^{\mathrm{T}}\mathbf{b}\end{aligned}$$

where we denote

$$\begin{aligned}R^{[k]} &= [(I - 1/L\,A^{\mathrm{T}}A)(\widehat{D}^{[k]})^2] \\ \mathbf{h}^{[k]} &= -(I - 1/L\,A^{\mathrm{T}}A)\lambda/L\,\widehat{\mathbf{d}}^{[k]} + 1/L\,A^{\mathrm{T}}\mathbf{b}\end{aligned} \tag{3.2.6}$$

throughout this chapter. Therefore, the ISTA in Alg. 1 with variable $\widehat{\mathbf{x}}$ can be modified to the following procedure using the new variables $\widehat{\mathbf{w}}$ and $\widehat{D}$.

Algorithm 9 One pass of modified ISTA

Input: Start with $\widehat{\mathbf{w}}^{[k]}$,$\widehat{D}^{[k]}$.

1: Set $\widehat{\mathbf{w}}^{[k+1]} = R^{[k]}\widehat{\mathbf{w}}^{[k]} + \mathbf{h}^{[k]}$ (with $R^{[k]}, \mathbf{h}^{[k]}$ defined by (3.2.6)).

2: Set $\widehat{D}^{[k+1]} = \textsc{Diag}(\mathrm{sign}\,(\mathrm{Shr}_{\lambda/L}(\widehat{\mathbf{w}}^{[k+1]})))$.

Output: $\widehat{\mathbf{w}}^{[k+1]}$, $\widehat{D}^{[k+1]}$ for next pass.

Algorithm 9 is mathematically equivalent to Algorithm 7 and is designed only for the purpose of analysis, not intended for computation. We note that step 1 of Alg. 5 can be written as a homogeneous matrix recurrence in (3.2.7), which we will use to characterize

ISTA's convergence.

$$\begin{aligned}\begin{pmatrix}\widehat{\mathbf{w}}^{[k+1]}\\ 1\end{pmatrix} &= \mathbf{R}_{aug}^{[k]}\begin{pmatrix}\widehat{\mathbf{w}}^{[k]}\\ 1\end{pmatrix} = \begin{pmatrix}R^{[k]} & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}\begin{pmatrix}\widehat{\mathbf{w}}^{[k]}\\ 1\end{pmatrix}\\ &= \begin{pmatrix}(I - {}^{1}\!/_{L} A^{\mathrm{T}}A)(\widehat{D}^{[k]})^2 & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}\begin{pmatrix}\widehat{\mathbf{w}}^{[k]}\\ 1\end{pmatrix}\end{aligned} \tag{3.2.7}$$

where we denote $\mathbf{R}_{aug}^{[k]}$ as $\begin{pmatrix}R^{[k]} & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}$, the augmented matrix of $R^{[k]}$, in this chapter.

It is known that the fixed point condition in Alg. 1 is equivalent to the KKT conditions (2.1.4) and (2.1.5). The following lemma shows the equivalence between the fixed point of the constructed matrix recurrence (3.2.7) and the KKT point of the LASSO problem.

Lemma 3.2.1 *Suppose* $\begin{pmatrix}\widehat{\mathbf{w}}\\ 1\end{pmatrix}$ *is an eigenvector corresponding to eigenvalue 1 of* $\mathbf{R}_{aug}$*(omitting* $[k]$*) in* (3.2.7) *and* $\widehat{D}^{[k+1]} = \widehat{D}^{[k]} = \widehat{D} = \textsc{Diag}(\widehat{\mathbf{d}})$ *with entries* $\widehat{d}_i = 1$ *if* $\widehat{w}_i > {}^{\lambda}\!/_{L}$, $\widehat{d}_i = -1$ *if* $\widehat{w}_i < -{}^{\lambda}\!/_{L}$ *and* $\widehat{d}_i = 0$ *if* $-{}^{\lambda}\!/_{L} \le \widehat{w}_i \le {}^{\lambda}\!/_{L}$. *Then the variable defined by* $\widehat{\mathbf{x}} = Shr_{{}^{\lambda}\!/_{L}}(\widehat{\mathbf{w}})$ *satisfies the* 1*st order KKT conditions* (2.1.4) *and* (2.1.5). *Conversely, if* $\widehat{\mathbf{x}}$ *and* $\widehat{\boldsymbol{\nu}} = {}^{1}\!/_{\lambda} A^{\mathrm{T}}(\mathbf{b} - A\widehat{\mathbf{x}})$ *satisfy the KKT conditions, then* $\begin{pmatrix}\widehat{\mathbf{w}}\\ 1\end{pmatrix}$, *with* $\widehat{\mathbf{w}} = \widehat{\mathbf{x}} + {}^{\lambda}\!/_{L}\widehat{\boldsymbol{\nu}}$, *is an eigenvector of* $\mathbf{R}_{\mathrm{aug}}$ *corresponding to eigenvalue 1, where* $\mathbf{R}_{aug}$ *is defined in* (3.2.7) *and* $\widehat{D}^{[k+1]} = \widehat{D}^{[k]} = \widehat{D} = \textsc{Diag}(\widehat{\mathbf{d}})$ *with entries* $\widehat{d}_i = 1$ *if* $\widehat{w}_i > {}^{\lambda}\!/_{L}$, $\widehat{d}_i = -1$ *if* $\widehat{w}_i < -{}^{\lambda}\!/_{L}$ *and* $\widehat{d}_i = 0$ *if* $-{}^{\lambda}\!/_{L} \le \widehat{w}_i \le {}^{\lambda}\!/_{L}$.

Proof. Define $\widehat{\boldsymbol{\nu}}^{[k]} = {}^{1}\!/_{\lambda} A^{\mathrm{T}}(\mathbf{b} - A\widehat{\mathbf{x}}^{[k]})$ for all k. Then from Algorithm 1, $\widehat{\mathbf{x}}^{[k+1]} = \mathrm{Shr}_{{}^{\lambda}\!/_{L}}(\widehat{\mathbf{w}}^{[k]}) = \mathrm{Shr}_{{}^{\lambda}\!/_{L}}(\widehat{\mathbf{x}}^{[k]} + {}^{\lambda}\!/_{L}\widehat{\boldsymbol{\nu}}^{[k]})$. $\widehat{\mathbf{w}}^{[k]}$ is a fixed point if and only if $\begin{pmatrix}\widehat{\mathbf{w}}^{[k]}\\ 1\end{pmatrix}$ is a eigenvector corresponding to eigenvalue 1. But it is a fixed point if and only if $\widehat{\mathbf{x}}^{[k]} = \mathrm{Shr}_{{}^{\lambda}\!/_{L}}(\widehat{\mathbf{x}}^{[k]} + {}^{\lambda}\!/_{L}\widehat{\boldsymbol{\nu}}^{[k]})$, which holds if and only if $\widehat{\mathbf{x}}^{[k]}, \widehat{\boldsymbol{\nu}}^{[k]}$ satisfy conditions (2.1.4)

(2.1.5). This last statement follows directly from the fact that $x = \text{Shr}_\xi(x+y)$ if and only if $y = \text{Thr}_\xi(x+y)$, where Thr_ξ is the threshold function defined in (4.1.4) satisfying $\text{Shr}_\xi + \text{Thr}_\xi = \text{Id}$ (the identity function). □

3.2.2 Spectral Properties of ISTA Operator

We give here some spectral properties of ISTA operator $\mathbf{R}_{aug}^{[k]}$ that will play key roles in our convergence analysis. We first recall some theory relating the spectral radius to the matrix norm.

Theorem 3.2.2 *Let $\sigma(M)$ denote the spectral radius of an arbitrary square real matrix M. Then we have the following:*

(a). $\sigma(M) \leq \|M\|_2$.

(b). *If $\|M\|_2 = \sigma(M)$ then for any eigenvalue λ such that $|\lambda| = \sigma(M)$, the algebraic and geometric multiplicities of λ are the same (all Jordan blocks for λ is 1×1). Such a matrix is said to be a member of Class M [65, 92].*

(c). *If a λ such that $|\lambda| = \sigma(M)$ has a Jordan block of dimension larger than 1 (the geometric multiplicity is strictly less than the algebraic multiplicity), then for any $\epsilon > 0$ there exists a matrix norm $\|\cdot\|_G$ such that $\sigma(M) < \|M\|_G \triangleq \|GMG^{-1}\|_\infty \leq \sigma(M) + \epsilon$.*

We refer readers to [11, 51, 65, 92] for the proof of the above theorem. We remark that (a) and (b) are true for any operator norm but here we need it only for the matrix 2-norm.

Lemma 3.2.3 *Regarding ISTA, there are three properties of $R^{[k]}$:*

(a). $\|R^{[k]}\|_2 = \|(I - {}^1\!/_L A^{\mathrm{T}} A)(\widehat{D}^{[k]})^2\|_2 \leq 1$.

(b). *All eigenvalues must lie in the interval $[0, 1]$.*

(c). *If there exists one or more eigenvalues equal to 1, then eigenvalue 1 must have a complete set of eigenvectors.*

Proof. We here omit the pass number $^{[k]}$ for simplicity.

(a). $\|R\|_2 = \|(I - {}^1\!/_L A^{\mathrm{T}}A)\widehat{D}^2\|_2 \leq \|(I - {}^1\!/_L A^{\mathrm{T}}A)\|_2 \|\widehat{D}^2\|_2 \leq 1$.

(b). All eigenvalues of R are the same as those of $R' = \widehat{D}(I - {}^1\!/_L A^{\mathrm{T}}A)\widehat{D}$. Noticing $L \geq \|A^{\mathrm{T}}A\|_2$, we obtain $\|R'\|_2 \leq \|\widehat{D}^2\|_2 \|(I - {}^1\!/_L A^{\mathrm{T}}A)\|_2 \leq 1$. In addition, R' is symmetric and a positive semidefinite matrix. Hence all eigenvalues of R' must lie in the interval $[0, 1]$. Hence so should those of R.

(c). Because $\sigma(R) = \|R\|_2 = 1$, this statement follows directly from Theorem 3.2.2.

□

3.2.3 FISTA as Matrix Recurrence

Recall the FISTA iteration for the LASSO problem is

$$\begin{cases} \mathbf{y}^{[k]} &= \widetilde{\mathbf{x}}^{[k-1]} + \frac{t^{[k-1]}-1}{t^{[k]}}(\widetilde{\mathbf{x}}^{[k-1]} - \widetilde{\mathbf{x}}^{[k-2]}) \\ \widetilde{\mathbf{x}}^{[k]} &= \mathrm{Shr}_{\lambda/L}((I - {}^1\!/_L A^{\mathrm{T}}A)\mathbf{y}^{[k]} + {}^1\!/_L A^{\mathrm{T}}\mathbf{b}) \\ t^{[k+1]} &= \frac{1+\sqrt{1+4t^{[k]\,2}}}{2}. \end{cases}$$

With the similar technique for ISTA, we use auxiliary variables $\widetilde{\mathbf{w}}^{[k]}, \widetilde{D}^{[k]}$ to replace variable $\widetilde{\mathbf{x}}^{[k]}$ for carrying the FISTA iterations. We set

$$\widetilde{\mathbf{w}}^{[k]} = (I - {}^1\!/_L A^{\mathrm{T}}A)\mathbf{y}^{[k]} + {}^1\!/_L A^{\mathrm{T}}\mathbf{b}. \tag{3.2.8}$$

Hence,

$$\widetilde{\mathbf{x}}^{[k+1]} = \mathrm{Shr}_{\lambda/L}(\widetilde{\mathbf{w}}^{[k]}) = (\widetilde{D}^{[k]})^2\widetilde{\mathbf{w}}^{[k]} - {}^\lambda\!/_L\widetilde{\mathbf{d}}^{[k]} \tag{3.2.9}$$

where for all k, the flag matrix $\widetilde{D}^{[k]} = \mathrm{diag}(\widetilde{\mathbf{d}}^{[k]})$, and the vector $\widetilde{\mathbf{d}}^{[k]}$ is defined elementwise as

$$\widetilde{d}_i^{[k]} = \mathrm{sign}\left(\mathrm{Shr}_{\lambda/L}\,\widetilde{w}_i^{[k]}\right) = \begin{cases} 1 & \text{if } \widetilde{w}_i^{[k]} > \lambda/L \\ 0 & \text{if } -\lambda/L \le \widetilde{w}_i^{[k]} \le \lambda/L \\ -1 & \text{if } \widetilde{w}_i^{[k]} < -\lambda/L. \end{cases} \tag{3.2.10}$$

Using (3.2.8), (3.2.9) and the updating formula in Alg. 2, we arrive at

$$\begin{aligned} \widetilde{\mathbf{w}}^{[k+1]} &= (I - 1/L A^{\mathrm{T}}A)\left[\widetilde{\mathbf{x}}^{[k]} + \tfrac{t^{[k]}-1}{t^{[k+1]}}(\widetilde{\mathbf{x}}^{[k]} - \widetilde{\mathbf{x}}^{[k-1]})\right] + 1/L A^{\mathrm{T}}\mathbf{b} \\ &= (I - 1/L A^{\mathrm{T}}A)\left[(\tfrac{t^{[k]}-1}{t^{[k+1]}} + 1)((\widetilde{D}^{[k]})^2\widetilde{\mathbf{w}}^{[k]} - \lambda/L\widetilde{\mathbf{d}}^{[k]})\right] \\ &\quad -(I - 1/L A^{\mathrm{T}}A)\left[\tfrac{t^{[k]}-1}{t^{[k+1]}}((\widetilde{D}^{[k-1]})^2\widetilde{\mathbf{w}}^{[k-1]} - \lambda/L\widetilde{\mathbf{d}}^{[k-1]})\right] + 1/L A^{\mathrm{T}}\mathbf{b} \\ &= (1 + \tfrac{t^{[k]}-1}{t^{[k+1]}})\left[(I - 1/L A^{\mathrm{T}}A)(\widetilde{D}^{[k]})^2\right]\widetilde{\mathbf{w}}^{[k]} \\ &\quad -\tfrac{t^{[k]}-1}{t^{[k+1]}}\left[(I - 1/L A^{\mathrm{T}}A)(\widetilde{D}^{[k-1]})^2\right]\widetilde{\mathbf{w}}^{[k-1]} \\ &\quad +(I - 1/L A^{\mathrm{T}}A)\left[-(1 + \tfrac{t^{[k]}-1}{t^{[k+1]}})\lambda/L\widetilde{\mathbf{d}}^{[k]} + \tfrac{t^{[k]}-1}{t^{[k+1]}}\lambda/L\widetilde{\mathbf{d}}^{[k-1]}\right] + 1/L A^{\mathrm{T}}\mathbf{b} \\ &= (1 + \tau^{[k]})\widetilde{R}^{[k]}\widetilde{\mathbf{w}}^{[k]} - \tau^{[k]}\widetilde{R}^{[k-1]}\widetilde{\mathbf{w}}^{[k-1]} + \widetilde{\mathbf{h}} \\ &= P^{[k]}\widetilde{\mathbf{w}}^{[k]} + Q^{[k-1]}\widetilde{\mathbf{w}}^{[k-1]} + \widetilde{\mathbf{h}}^{[k]} \end{aligned} \tag{3.2.11}$$

where we denote

$$\begin{cases} \tau^{[k]} &= \frac{t^{[k]}-1}{t^{[k+1]}} \\ P^{[k]} &= (1 + \tau^{[k]})\widetilde{R}^{[k]} \\ Q^{[k]} &= -\tau^{[k+1]}\widetilde{R}^{[k]} \\ \widetilde{R}^{[k]} &= (I - 1/L A^{\mathrm{T}}A)(\widetilde{D}^{[k]})^2 \\ \widetilde{\mathbf{h}}^{[k]} &= (I - 1/L A^{\mathrm{T}}A)\left[-(1 + \tau^{[k]})\lambda/L\widetilde{\mathbf{d}}^{[k]} + \tau^{[k]}\lambda/L\widetilde{\mathbf{d}}^{[k-1]}\right] + 1/L A^{\mathrm{T}}\mathbf{b} \end{cases} \tag{3.2.12}$$

in the rest of this chapter. Note that $R^{[k]}$ in (3.2.6) refers to the mapping at the k-th iteration of ISTA while $\widetilde{R}^{[k]}$ in (3.2.12) refers to the mapping that would occur if one took one step of ISTA starting at the k-th iterate of FISTA. Note also that if $\widetilde{\mathbf{d}}^{[k]} = \widetilde{\mathbf{d}}^{[k-1]}$, then $\widetilde{\mathbf{h}}^{[k]} = (I - {}^1\!/_L A^{\mathrm{T}} A)\left[{}^\lambda\!/_L \widetilde{\mathbf{d}}^{[k]}\right] + {}^1\!/_L A^{\mathrm{T}}\mathbf{b}$ does not vary with τ. For the purposes of analysis, the modified FISTA iteration then can be equivalently expressed as in Algorithm 10 .

Algorithm 10 One pass of modified FISTA

Input: Start with $\widetilde{\mathbf{w}}^{[k-1]}$, $\widetilde{\mathbf{w}}^{[k]}$, $t^{[k]}$, $\widetilde{D}^{[k-1]}$ and $\widetilde{D}^{[k]}$.

1: Set $\widetilde{\mathbf{w}}^{[k+1]} = P^{[k]}\widetilde{\mathbf{w}}^{[k]} + Q^{[k-1]}\widetilde{\mathbf{w}}^{[k-1]} + \widetilde{\mathbf{h}}^{[k]}$ (with $P^{[k]}, Q^{[k-1]}, \widetilde{\mathbf{h}}^{[k]}$ defined by (3.2.12)).

2: Set $t^{[k+1]} = \frac{1+\sqrt{1+4t^{[k]2}}}{2}$ so that $\tau^{[k]} = \frac{t^{[k]}-1}{t^{[k+1]}}$.

3: Set $\widetilde{D}^{[k+1]} = \textsc{Diag}(\mathrm{sign}\,(\mathrm{Shr}_{{}^\lambda\!/_L}(\widetilde{\mathbf{w}}^{[k+1]})))$.

Output: $\widetilde{\mathbf{w}}^{[k]}$, $\widetilde{\mathbf{w}}^{[k+1]}$, $t^{[k+1]}$, $\widetilde{D}^{[k]}$ and $\widetilde{D}^{[k+1]}$ for next pass.

Step 1 of Algorithm 10 can also be formulated as a homogeneous matrix recurrence analogous to (3.2.7) for ISTA, but with a larger (approximately double) dimension:

$$\begin{aligned}\begin{pmatrix} \widetilde{\mathbf{w}}^{[k+1]} \\ \widetilde{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix} &= \mathbf{N}_{aug}^{[k]} \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} = \begin{pmatrix} N^{[k]} & \widetilde{\mathbf{h}}_{aug}^{[k]} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} P^{[k]} & Q^{[k-1]} & \widetilde{\mathbf{h}}^{[k]} \\ I & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix}.\end{aligned} \tag{3.2.13}$$

We denote $N^{[k]} = \begin{pmatrix} P^{[k]} & Q^{[k-1]} \\ I & 0 \end{pmatrix}$ and $\widetilde{\mathbf{h}}_{aug}^{[k]} = \begin{pmatrix} \widetilde{\mathbf{h}}^{[k]} \\ 0 \end{pmatrix}$ such that $\mathbf{N}_{aug}^{[k]} = \begin{pmatrix} N^{[k]} & \widetilde{\mathbf{h}}_{aug}^{[k]} \\ 0 & 1 \end{pmatrix}$ in the remainder of this chapter.

Analogous to Lemma 3.2.1 for ISTA, the following lemma shows the equivalence between the fixed point of the constructed matrix recurrence (3.2.13) and the KKT point

of the LASSO problem.

Lemma 3.2.4 *Suppose* $\begin{pmatrix} \widetilde{\mathbf{w}}_1 \\ \widetilde{\mathbf{w}}_2 \\ 1 \end{pmatrix}$ *is an eigenvector corresponding to eigenvalue 1 of* $\mathbf{N}_{aug}$ *(omitting* $[k]$*) in* (3.2.13)*, then* $\widetilde{\mathbf{w}}_1 = \widetilde{\mathbf{w}}_2 := \widetilde{\mathbf{w}}$*. Suppose further that* $\widetilde{D}^{[k+1]} = \widetilde{D}^{[k]} = \widetilde{D} = \textsc{Diag}(\widetilde{\mathbf{d}})$ *with entries* $\widetilde{d}_i = 1$ *if* $\widetilde{w}_i > \lambda/L$*,* $\widetilde{d}_i = -1$ *if* $\widetilde{w}_i < -\lambda/L$ *and* $\widetilde{d}_i = 0$ *if* $-\lambda/L \leq \widetilde{w}_i \leq \lambda/L$*. Then the variables defined by* $\widetilde{\mathbf{x}} = Shr_{\lambda/L}(\widetilde{\mathbf{w}})$ *satisfies the* 1*st order KKT conditions* (2.1.4) *and* (2.1.5)*. Conversely, if* $\widetilde{\mathbf{x}}$ *and* $\widetilde{\boldsymbol{\nu}} = 1/\lambda A^{\mathrm{T}}(\mathbf{b} - A\widetilde{\mathbf{x}})$ *satisfy the KKT conditions, then* $\begin{pmatrix} \widetilde{\mathbf{w}} \\ \widetilde{\mathbf{w}} \\ 1 \end{pmatrix}$*, with* $\widetilde{\mathbf{w}} = \widetilde{\mathbf{x}} + \lambda/L\widetilde{\boldsymbol{\nu}}$*, is an eigenvector of* $\mathbf{N}_{aug}$ *corresponding to eigenvalue 1, where* $\mathbf{N}_{aug}$ *is defined in* (3.2.13) *and* $\widetilde{D}^{[k+1]} = \widetilde{D}^{[k]} = \widetilde{D} = \textsc{Diag}(\widetilde{\mathbf{d}})$ *with entries* $\widetilde{d}_i = 1$ *if* $\widetilde{w}_i > \lambda/L$*,* $\widetilde{d}_i = -1$ *if* $\widetilde{w}_i < -\lambda/L$ *and* $\widetilde{d}_i = 0$ *if* $-\lambda/L \leq \widetilde{w}_i \leq \lambda/L$*.*

Proof. It is easy to show that a vector is a fixed point for FISTA if and only if it is a fixed point for ISTA, so this follows from Lemma 3.2.1. □

To prepare for further discussion, we make three remarks.

(a). $\tau^{[k]} \longrightarrow 1$ from below as $k \longrightarrow \infty$.

(b). $R^{[k]} = \widetilde{R}^{[k]}$ if the flag matrix of ISTA and FISTA are the same, i.e. $\widehat{D}^{[k]} = \widetilde{D}^{[k]}$ and $\mathbf{h}^{[k]} = \widetilde{\mathbf{h}}^{[k]}$ if $\widehat{D}^{[k]} = \widetilde{D}^{[k]} = \widetilde{D}^{[k-1]}$. This observation relates the $\mathbf{R}_{aug}^{[k]}$ to $\mathbf{N}_{aug}^{[k]}$. It is the foundation upon which we establish the properties to compare ISTA and FISTA in Section 3.5.

(c). One main difference between operators of ISTA and FISTA (i.e. $\mathbf{R}_{aug}^{[k]}$ and $\mathbf{N}_{aug}^{[k]}$) is that $\mathbf{R}_{aug}^{[k]}$ is fixed when the flag matrix is fixed while $\mathbf{N}_{aug}^{[k]}$ changes at each step k even if the flag matrix is fixed. In other words, for all k, $\mathbf{R}_{aug}^{[k]} = \mathbf{R}_{aug}^{[k+1]}$ if $\widehat{D}^{[k]} = \widehat{D}^{[k+1]}$. But $\mathbf{N}_{aug}^{[k]} \neq \mathbf{N}_{aug}^{[k+1]}$ even if $\widetilde{D}^{[k]} = \widetilde{D}^{[k+1]}$. The reason is that $\mathbf{N}_{aug}^{[k]}$ depends on the changing stepsize $\tau^{[k]}$. Nevertheless, one can still use the same similar argument for $\mathbf{N}_{aug}^{[k]}$ as for

$\mathbf{R}_{aug}^{[k]}$ by additional lemmas, as we will show in Section 3.4.

3.2.4 Spectral Properties of FISTA Operator

We give the spectral properties of the FISTA matrix operator that we will use in our convergence analysis. Lemma 3.2.5 gives the eigenstructure of $N^{[k]}$ and its relation to that of the ISTA operator $\widetilde{R}^{[k]}$.

Lemma 3.2.5 *Suppose $\widetilde{D}^{[k]} = \widetilde{D}^{[k-1]}$ and hence $\widetilde{R}^{[k]} = \widetilde{R}^{[k-1]}$, then (omitting index $^{[k]}$) we have the following results:*

(a). *For any given eigenvalue γ of N, it must have a corresponding eigenvector with the form of $\begin{pmatrix} \gamma\widetilde{\mathbf{w}} \\ \widetilde{\mathbf{w}} \end{pmatrix}$. And for that given γ, there exists an eigenvalue of $\widetilde{R}$, $\beta = \frac{\gamma^2}{[(1+\tau)\gamma-\tau]}$ with corresponding eigenvector $\widetilde{\mathbf{w}}$. Conversely, let β be any eigenvalue of $\widetilde{R}$ with corresponding eigenvector $\widetilde{\mathbf{w}}$. Then for given β, there exists a pair of eigenvalues of N, γ_1 and γ_2, which are the solutions of $\gamma^2 - \gamma(1+\tau)\beta + \tau\beta = 0$. Furthermore, $\begin{pmatrix} \gamma_1\widetilde{\mathbf{w}} \\ \widetilde{\mathbf{w}} \end{pmatrix}$ and $\begin{pmatrix} \gamma_2\widetilde{\mathbf{w}} \\ \widetilde{\mathbf{w}} \end{pmatrix}$ are two corresponding eigenvectors of N.*

(b). *For $0 < \tau \leq 1$, the eigenvalues of N defined in* (3.2.13) *lie in the closed disk in the complex plane with center $1/2$ and radius $1/2$, denoted as $\mathcal{D}(1/2, 1/2)$. In particular, if N has any eigenvalue with absolute value $\sigma(N) = 1$, then that eigenvalue must be exactly 1.*

(c). *N has an eigenvalue equal to 1 if and only if $\widetilde{R}$ has an eigenvalue equal to 1.*

(d). *Assuming $\tau < 1$, then if N has an eigenvalue equal to 1, this eigenvalue must have a complete set of eigenvectors.*

Proof. (a). Given any eigenvalue γ of N, by definition (just after (3.2.13))

$$N \cdot \begin{pmatrix} \widetilde{\mathbf{w}}_1 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix} = \begin{pmatrix} P & Q \\ I & 0 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{w}}_1 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix} = \begin{pmatrix} P\widetilde{\mathbf{w}}_1 + Q\widetilde{\mathbf{w}}_2 \\ \widetilde{\mathbf{w}}_1 \end{pmatrix} = \gamma \begin{pmatrix} \widetilde{\mathbf{w}}_1 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix}$$

and thus $\widetilde{\mathbf{w}}_1 = \gamma\widetilde{\mathbf{w}}_2$ is observed from the second row. Then

$$N \cdot \begin{pmatrix} \gamma\widetilde{\mathbf{w}}_2 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix} = \begin{pmatrix} \gamma P\widetilde{\mathbf{w}}_2 + Q\widetilde{\mathbf{w}}_2 \\ \gamma\widetilde{\mathbf{w}}_2 \end{pmatrix} = \begin{pmatrix} (1+\tau)\gamma\widetilde{R}\widetilde{\mathbf{w}}_2 - \tau\widetilde{R}\widetilde{\mathbf{w}}_2 \\ \gamma\widetilde{\mathbf{w}}_2 \end{pmatrix} = \gamma \begin{pmatrix} \gamma\widetilde{\mathbf{w}}_2 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix}.$$

Therefore,

$$\widetilde{R}\widetilde{\mathbf{w}}_2 = \frac{\gamma^2}{[(1+\tau)\gamma - \tau]}\widetilde{\mathbf{w}}_2 = \beta\widetilde{\mathbf{w}}_2 \iff \gamma^2 - (1+\tau)\gamma\beta + \tau\beta = 0. \tag{3.2.14}$$

(b). We first study the spectrum of matrix $N - \frac{1}{2}I$, then the spectrum of N should be a shift by $\frac{1}{2}$. Let $\alpha = \gamma - \frac{1}{2}$ be the eigenvalue of $N - \frac{1}{2}I$ associated with eigenvector $\begin{pmatrix} \widetilde{\mathbf{w}}_1 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix}$, then according to (3.2.14), α and β satisfy

$$\alpha^2 + (1 - \beta - \tau\beta)\alpha + \tfrac{1}{2}\tau\beta - \tfrac{1}{2}\beta + \tfrac{1}{4} = 0.$$

Note that $\tau \in (0, 1]$ and $\beta \in [0, 1]$ by definition of $\widetilde{R}$. We consider two situations for the above quadratic equation. First, suppose α_1 and α_2 are two conjugate complex roots. Then $\alpha_1 = \bar{\alpha_2}$, $\alpha_1 + \alpha_2 = \tau\beta + \beta - 1$ and $\alpha_1\alpha_2 = \frac{1}{2}\tau\beta - \frac{1}{2}\beta + \frac{1}{4}$ such that

$$|\alpha|^2 = |\alpha_1\bar{\alpha_1}| = |\alpha_1\alpha_2| = \left|\tfrac{1}{2}\tau\beta - \tfrac{1}{2}\beta + \tfrac{1}{4}\right| \leq \frac{1}{4} \tag{3.2.15}$$

which gives $|\alpha| \leq \frac{1}{2}$. The second situation is that both roots are real numbers and they are

$$\alpha_1 = \frac{1+\tau}{2}\beta + \frac{\sqrt{\beta}\sqrt{(1+\tau)^2\beta - 4\tau}}{2} - \tfrac{1}{2} \quad \alpha_2 = \frac{1+\tau}{2}\beta - \frac{\sqrt{\beta}\sqrt{(1+\tau)^2\beta - 4\tau}}{2} - \tfrac{1}{2}.$$

To get the largest possible value of α, we only look at α_1 because $\alpha_1 \geq \alpha_2$ for any

fixed β. Since α_1 is an increasing function of β and $\beta \in [0,1]$, the largest real value of α should be $1/2$ when $\beta = 1$. On the other hand, to get the smallest negative real value of α, we only need to look at α_2. One can write $\alpha_2 = \frac{1+\tau}{2}(\beta - \sqrt{\beta^2 - \frac{4\tau}{(1+\tau)^2}}) - 1/2$ to see that $\alpha_2 \geq -1/2$. So we conclude that if α is real, then $-1/2 \leq \alpha \leq 1/2$.

Under both situations, one can conclude that α must satisfy $|\alpha| \leq 1/2$, lying in a disk centered at 0 with radius $1/2$, i.e. $\mathcal{D}(0, 1/2)$. So the eigenvalues of N must lie in the disk $\mathcal{D}(1/2, 1/2)$ by the shift. Consequently, the only possible eigenvalue on the unit circle is 1.

(c). γ_1, γ_2 are the two roots of the quadratic polynomial, i.e. $\gamma^2 - (1+\tau)\gamma\beta + \tau\beta = (\gamma - \gamma_1)(\gamma - \gamma_2) = 0$. For given β, they must satisfy

$$\gamma_1\gamma_2 = \tau\beta \quad \text{and} \quad \gamma_1 + \gamma_2 = (1+\tau)\beta = \beta + \gamma_1\gamma_2.$$

If N has an eigenvalue $\gamma_1 = 1$, then $\gamma_2 = (1+\tau)\beta - 1 = \beta + \gamma_1\gamma_2 - \gamma_1 = \beta + \gamma_2 - 1$, hence $\beta = 1$ must be true and $\widetilde{R}$ has an eigenvalue equal to 1. Conversely, if $\widetilde{R}$ has an eigenvalue $\beta = 1$, the quadratic polynomial (3.2.14) will reduce to $\gamma^2 - (1+\tau)\gamma + \tau = 0$, which gives $\gamma_1 = 1$ and $\gamma_2 = \tau$. Then N has an eigenvalue equal to 1.

(d). Notice in (3.2.14) that each eigenvalue β of $\widetilde{R}$ maps to two eigenvalues of N, γ_1 and γ_2, and associated eigenvector $\widetilde{\mathbf{w}}_2$ of $\widetilde{R}$ maps to two eigenvectors of N, $\begin{pmatrix} \gamma_1\widetilde{\mathbf{w}}_2 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix}$ and $\begin{pmatrix} \gamma_2\widetilde{\mathbf{w}}_2 \\ \widetilde{\mathbf{w}}_2 \end{pmatrix}$. As shown in (c), N has an eigenvalue equal to 1 if and only if $\widetilde{R}$ has an eigenvalue equal to 1. Since R and $\widetilde{R}$ have the similar eigenstructure, eigenvalue 1 of $\widetilde{R}$ must have a complete set of eigenvectors. So the only possible situation that eigenvalue 1 of N does not have a complete set of eigenvectors is that both γ_1 and γ_2 equal to 1. However, this is impossible because we have shown in (c) that $\beta = 1$ gives $\gamma_1 = 1$ and $\gamma_2 = \tau$ which is close to 1 but not equal. As a result, if N has an eigenvalue 1, and then its algebraic and geometric multiplicities coincide. □

3.3 Regimes

Since the ISTA and FISTA updating steps have been converted into a variation of an eigenproblem in previous sections, we can study the convergence in terms of the spectral properties of the operator $\mathbf{R}_{aug}^{[k]}$ in (3.2.7) and $\mathbf{N}_{aug}^{[k]}$ in (3.2.13). Hence in this section, we show how these properties reflected in the possible convergence "regimes" that ISTA and FISTA can encounter.

3.3.1 Spectral Properties

The eigenvalues of the augmented matrices $\mathbf{R}_{aug}^{[k]}$ and $\mathbf{N}_{aug}^{[k]}$ consist of those of $R^{[k]}$, $N^{[k]}$, respectively, plus an extra eigenvalue equal to 1. If $R^{[k]}$ or $N^{[k]}$ already has an eigenvalue equal to 1, then the extra eigenvalue 1 may or may not add a corresponding eigenvector. The next lemma gives limits on the properties of the eigenalue 1 for any augmented matrix of the general form of $\mathbf{R}_{aug}^{[k]}$, $\mathbf{N}_{aug}^{[k]}$.

Lemma 3.3.1 *Let* $\mathbf{M}_{aug} = \begin{pmatrix} M & \mathbf{p} \\ 0 & 1 \end{pmatrix}$ *be any block upper triangular matrix with a* 1×1 *lower right block. The matrix* $\mathbf{M}_{aug}$ *has an eigenvalue* $\alpha_1 = 1$ *and suppose its corresponding eigenvector has a non-zero last element. We scale that eigenvector to take the form* $\begin{pmatrix} \mathbf{w} \\ 1 \end{pmatrix} = \mathbf{M}_{aug} \begin{pmatrix} \mathbf{w} \\ 1 \end{pmatrix}$. *If the upper left block* M *either has no eigenvalue equal to* 1 *or the eigenvalue* 1 *of* M *has a complete set of eigenvectors, then* $\alpha_1 = 1$ *has no non-trivial Jordan block. Furthermore, if the given eigenvector* $\begin{pmatrix} \mathbf{w} \\ 1 \end{pmatrix}$ *is unique, then* M *has no eigenvalue equal to 1.*

We refer readers to [11] for the proof of Lemma 3.3.1. Now we summarize spectral properties of our specific operators $\mathbf{R}_{aug}^{[k]}$ and $\mathbf{N}_{aug}^{[k]}$ in terms of their possible Jordan canonical forms as given in the following lemmas. Essentially these lemmas say that all

their eigenvalues must have absolute value strictly less than 1, except for the eigenvalue equal to 1. And the eigenvalue 1's geometric multiplicity either equal to or one less than its algebraic multiplicity.

Lemma 3.3.2 *Assuming* $\widehat{D}^{[k+1]} = \widehat{D}^{[k]}$*, then* $\mathbf{R}_{\text{aug}}^{[k]}$ *in* (3.2.7) *is fixed and has a spectral decomposition* $\mathbf{R}_{\text{aug}}^{[k]} = \mathbf{P}_R \mathbf{J}_R^{[k]} \mathbf{P}_R^{-1}$*, where* $\mathbf{J}_R^{[k]}$ *is a block diagonal matrix*

$$\mathbf{J}_R^{[k]} = \textsc{Diag}\left\{ \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, I_R^{[k]}, \widehat{\mathbf{J}}_R^{[k]} \right\} \tag{3.3.16}$$

where any of these blocks might be missing. Here $I_R^{[k]}$ *is an identity matrix and* $\widehat{\mathbf{J}}_R^{[k]}$ *is a matrix with spectral radius strictly less than* 1*.*

Proof. For $\mathbf{R}_{\text{aug}}^{[k]}$, the upper left block of (3.2.7) (i.e. $R^{[k]}$) satisfies Lemma 3.2.3 and hence contributes blocks of the form $I_R^{[k]}$, $\widehat{\mathbf{J}}_R^{[k]}$. No eigenvalue with absolute value 1 can have a nondiagonal Jordan block, so the blocks corresponding to those eigenvalues must be diagonal. Embedding that upper left block $R^{[k]}$ into the entire matrix yields a matrix $\mathbf{R}_{aug}^{[k]}$, with the exact same set of eigenvalues with the same algebraic and geometric multiplicities, except for eigenvalue 1.

If the upper left block of $\mathbf{R}_{aug}^{[k]}$ has no eigenvalue equal to 1, then $\mathbf{R}_{aug}^{[k]}$ has a simple eigenvalue 1. In general for eigenvalue 1, the algebraic multiplicity goes up by one and the geometric multiplicity can either stay the same or increase by 1. In other words, $\mathbf{R}_{aug}^{[k]}$ either satisfies the conditions of Lemma 3.3.1, or the algebraic and geometric multiplicities of eigenvalue 1 for $\mathbf{R}_{aug}^{[k]}$ differ by 1, meaning we have a single 2×2 Jordan block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

□

Lemma 3.3.3 *Assuming* $\widetilde{D}^{[k+1]} = \widetilde{D}^{[k]}$*, since* $\mathbf{N}_{aug}^{[k]}$ *in* (3.2.13) *is different at each step, for each step* k*, there exists a* $\mathbf{P}_N^{[k]}$ *such that* $\mathbf{N}_{aug}^{[k]}$ *has a spectral decomposition*

$\mathbf{N}_{aug}^{[k]} = \mathbf{P}_N^{[k]}\mathbf{J}_N^{[k]}(\mathbf{P}_N^{[k]})^{-1}$, *where* $\mathbf{J}_N^{[k]}$ *is the block diagonal matrix:*

$$\mathbf{J}_N^{[k]} = \textsc{Diag}\left\{ \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, I_N^{[k]}, \widetilde{\mathbf{J}}_N^{[k]} \right\} \tag{3.3.17}$$

where any of these blocks might be missing. Here $I_N^{[k]}$ *is an identity matrix,* $\widetilde{\mathbf{J}}_N^{[k]}$ *is a matrix with spectral radius strictly less than* 1*.*

Proof. The proof is similar to $\mathbf{R}_{aug}^{[k]}$. We only note here that the upper left block of (3.2.13) (i.e. $N^{[k]}$) satisfies Lemma 3.2.5 and hence contributes blocks of the form $I_N^{[k]}$, $\widetilde{\mathbf{J}}_N^{[k]}$. □

3.3.2 Four Types of Regimes

Lemmas 3.3.2 & 3.3.3 give rise to the four possible "regimes" associated with the ISTA and FISTA iterations, depending on the flag matrix and the eigenvalues of $\mathbf{R}_{aug}^{[k]}$, $\mathbf{N}_{aug}^{[k]}$. We treat separately the case where the flag matrix remains the same at each iteration, in which there are three possible regimes, and treat all the transitional cases together in their own fourth regime. For simplicity of the notation, we let D denote the flag matrix instead of $\widehat{D}$ and $\widetilde{D}$ unless specified.

When the flag matrix does not change, i.e. $D^{[k+1]} = D^{[k]}$, the ISTA operator $\mathbf{R}_{aug}^{[k]}$ remains invariant over those passes while the FISTA operator $\mathbf{N}_{aug}^{[k]}$ is slightly different at each iteration due to the changing parameter $\tau^{[k]}$. In both cases, the convergence behavior of the algorithm should depend on the eigenvalue of its corresponding operator, which can be categorized into three situations. In the following, we specifically describe these three possible regimes distinguished by the eigenstructure of the operators $\mathbf{R}_{aug}^{[k]}$, $\mathbf{N}_{aug}^{[k]}$. One of the these three regimes must occur when the flag matrix is unchanged from one step to the next: $D^{[k+1]} = D^{[k]}$.

[A]. The spectral radius of $R^{[k]}$ (or $N^{[k]}$) is strictly less than 1. The block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ is absent from (3.3.16) (or (3.3.17)), and the block $I_R^{[k]}$ (or $I_N^{[k]}$) is 1×1. In the case where the optimal solution exists and is unique, the result is linear convergence to the solution when close enough to that solution, as we will show in Theorem 3.4.3 & 3.4.5.

For $\mathbf{R}_{aug}^{[k]}$, as long as the flags remain the same, the recurrence (3.2.7) hence will converge linearly to a unique fixed point at a rate determined by the next largest eigenvalue in absolute value (largest eigenvalue of the block $\widehat{\mathbf{J}}_R^{[k]}$), according to the theory of the power method. If the flags $\widehat{D}^{[k]}$ are consistent with the eigenvector satisfying (3.2.4), then the eigenvector must satisfy the KKT conditions (2.1.4) and (2.1.5) because of Lemma 3.2.1.

For $\mathbf{N}_{aug}^{[k]}$, though it changes slightly at each step, we will show in the case of a unique solution that the left and right eigenvectors for eigenvalue 1 do not depend on τ (Lemma 3.4.4), and the remaining eigenvalues remain smaller and bounded away from 1. The result is that we observe linear convergence to a unique fixed point with a slightly changing convergence rate. If the flags $\widetilde{D}^{[k]}$ are consistent with the eigenvector satisfying (3.2.10), then that eigenvector must satisfy the KKT conditions because of Lemma 3.2.4.

[B]. $R^{[k]}$ (or $N^{[k]}$) has an eigenvalue equal to 1 which results in a 2×2 Jordan block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ for $\mathbf{R}_{aug}^{[k]}$ (or $\mathbf{N}_{aug}^{[k]}$). Therefore, the iteration process tends to a constant step.

For $\mathbf{R}_{aug}^{[k]}$, the theory of power method implies that the vector iterates will converge to the invariant subspace corresponding to the largest eigenvalue 1. The presence of $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ means that there is a Jordan chain: two non-zero vectors $\mathbf{q}, \mathbf{r}$ such that $(\mathbf{R}_{aug} - I)\mathbf{q} = \mathbf{r}$, $(\mathbf{R}_{aug} - I)\mathbf{r} = 0$. Any vector which includes a component of the form $\alpha\mathbf{q} + \beta\mathbf{r}$ will be transformed into $\mathbf{R}_{aug}(\alpha\mathbf{q} + \beta\mathbf{r}) = \alpha\mathbf{q} + (\alpha + \beta)\mathbf{r}$, i.e. each pass would add a constant vector $\alpha\mathbf{r}$, plus fading lower order terms from the other lesser eigenvalues. As long as the

flags do not change, this will result in constant steps: the difference between consecutive iterates, $\begin{pmatrix} \widehat{\mathbf{w}}^{[k+1]} \\ 1 \end{pmatrix} - \begin{pmatrix} \widehat{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix}$, will converge to a constant vector, asymptotically as the effects of the smaller eigenvalues fade. That constant vector is an eigenvector for eigenvalue 1. The ISTA iteration will not converge until a flag change in $\widehat{\mathbf{w}}^{[k]}$ forces a change in the flags $\widehat{D}^{[k]}$. If we satisfy the conditions for global convergence of ISTA, then such a flag change is guaranteed to occur.

The same analysis applies to $\mathbf{N}_{aug}^{[k]}$. The difference between two consecutive iterates, $\begin{pmatrix} \widetilde{\mathbf{w}}^{[k+1]} \\ 1 \end{pmatrix} - \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix}$, will asyptotically converge to a constant vector. The FISTA iteration will not converge until a flag change in $\widetilde{\mathbf{w}}^{[k]}$ forces a change in the flags $\widetilde{D}^{[k]}$. Such a flag change is guaranteed to occur due to the global convergence of FISTA under the assumption of unique solution. In Section 3.5, we will show that FISTA can jump out of such regime very fast, which is the main reason why it is faster than ISTA. See Section 3.6 for more discussions on its numerical behavior.

[C]. $R^{[k]}$ (or $N^{[k]}$) has an eigenvalue equal to 1, but the block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ is absent. For $\mathbf{R}_{aug}^{[k]}$ (or $\mathbf{N}_{aug}^{[k]}$), the convergence rate of this regime will still depend on $\sigma(\widehat{\mathbf{J}}_R^{[k]})$ and $\sigma(\widetilde{\mathbf{J}}_N^{[k]})$. If we assume the solution to be unique, the eigenvalue 1 of $\mathbf{R}_{aug}^{[k]}$ (or $\mathbf{N}_{aug}^{[k]}$) must be simple by Lemma 3.3.1. So the iteration will eventually jump out of this type of regime.

When the flag matrix does change, it means the set of active constraints at the current pass in the process has changed, and the current pass is a transition to a different operator with a different eigenstructure.

[D]. The operator $R^{[k+1]}$ (or $N^{[k+1]}$) will be different from $R^{[k]}$ (or $N^{[k]}$) due to different flag matrix.

3.4 Local Linear Convergence

In the case that the LASSO problem (2.1.3) has a unique solution with strict complementarity (Section 2.1.2), we can guarantee that eventually the flag matrix will not change.

Once the flag matrix stays fixed, the ISTA (or FISTA) iteration behaves just like the power method (or similar to power method) for the matrix eigenvalue problem. In this case, the spectral theory developed here gives a guarantee of linear convergence with the rate equal to the second largest eigenvalue of the matrix operator. In this section, we denote the fixed point of the respective matrix recursions (3.2.7) and (3.2.13) as

$$\widehat{\mathbf{w}}^*_{aug} = \begin{pmatrix} \widehat{\mathbf{w}}^* \\ 1 \end{pmatrix} \text{ and } \widetilde{\mathbf{w}}^*_{aug} = \begin{pmatrix} \widetilde{\mathbf{w}}^* \\ \widetilde{\mathbf{w}}^* \\ 1 \end{pmatrix}, \tag{3.4.18}$$

where in the final regime, $\widehat{\mathbf{w}}^* = \widetilde{\mathbf{w}}^* = \mathbf{w}^* = \mathbf{x}^* + {}^{\lambda}\!/_{L}\boldsymbol{\nu}^*$, with $\mathbf{x}^*, \boldsymbol{\nu}^*$ the unique solution satisfying (2.1.4) and (2.1.5).

3.4.1 Global Convergence Theory

We first invoke the global convergence property of ISTA and FISTA.

Theorem 3.4.1 *If problem* (2.1.3) *is solvable, let* $F^* = \min_{\mathbf{x}} F(\mathbf{x})$*, where* $F(\mathbf{x}) = {}^{1}\!/_{2}\|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1$*. Let* $\widehat{\mathbf{x}}^{[0]}$ *be any starting point in* $\mathbb{R}^p$ *and* $\widehat{\mathbf{x}}^{[k]}$ *be the sequence generated by ISTA. Then for any* $k \geq 1$*,* $F(\widehat{\mathbf{x}}^{[k]}) - F^*$ *decreases to zero at a rate bounded by* $O({}^{1}\!/_{k})$*. On the other hand, if we let* $\mathbf{y}^{[1]} = \widetilde{\mathbf{x}}^{[0]}$ *be any starting point in* $\mathbb{R}^p$*,* $t^{[0]} = t^{[1]} = 1$ *and* $\{\widetilde{\mathbf{x}}^{[k]}\}, \{\mathbf{y}^{[k]}\}, \{t^{[k]}\}$ *be the sequence generated by FISTA, then for any* $k \geq 1$*,* $F(\widetilde{\mathbf{x}}^{[k]}) - F^*$ *decreases to zero at a rate bounded by* $O({}^{1}\!/_{k^2})$*.*

This is a restatement of the convergence theorem in [9]. It says little on the local behavior of the algorithm. Under the assumption of unique solution with strict complementarity, we can prove the specific result that ISTA and FISTA iterations must eventually reach and remain in "linear convergence" regime [A] so that the optimal flag matrix is identified.

Lemma 3.4.2 *Suppose the LASSO problem* (2.1.3) *has a unique solution and this solution has strict complementarity (§2.1.2). Then eventually the ISTA (FISTA) iteration reaches and remains in the final regime where optimal flag matrix is identified.*

Proof. For ISTA, by Lemma 3.2.1, the strict complementarity is equivalent to $\widehat{w}_i^* \neq \pm\lambda/L$. Note that $\widehat{w}_i^{[k]}$ (where k is the pass number) could only be in one of three cases: $\widehat{w}_i^{[k]} > \lambda/L$, $\widehat{w}_i^{[k]} < -\lambda/L$ or $-\lambda/L \leq \widehat{w}_i^{[k]} \leq \lambda/L$. We can let $C_1 = \min\{|\widehat{w}_i^* - \lambda/L| - \epsilon_1, |\widehat{w}_i^* + \lambda/L| - \epsilon_1\} > 0$ for a positive constant ϵ_1 sufficiently small to make $C_1 > 0$. We define the following ball around eigenvector $\widehat{\mathbf{w}}_{aug}^*$:

$$\mathcal{B}_1 = \{\widehat{\mathbf{w}}_{aug} : \|\widehat{\mathbf{w}}_{aug} - \widehat{\mathbf{w}}_{aug}^*\|_\infty \leq C_1\}. \tag{3.4.19}$$

By Theorem 3.4.1 and uniqueness of the solution, the iterates $\widehat{\mathbf{w}}_{aug}^{[k]}$ converge to the $\widehat{\mathbf{w}}_{aug}^*$. Therefore, there exists a pass K_1 such that $\widehat{\mathbf{w}}_{aug}^{[k]}$ lies in $\mathcal{B}_1$ (i.e. $\|\widehat{\mathbf{w}}_{aug}^{[k]} - \widehat{\mathbf{w}}_{aug}^*\|_\infty \leq C_1$) for all $k > K_1$. This means that $\widehat{w}_i^{[k]}$ will remain in one of three cases: $\widehat{w}_i^{[k]} > \lambda/L$, $\widehat{w}_i^{[k]} < -\lambda/L$ or $-\lambda/L \leq \widehat{w}_i^{[k]} \leq \lambda/L$ and will never change to another case for all $k > K_1$. This, combined with the definition of flag matrix $\widehat{D}^{[k]} = \textsc{Diag}(\text{sign}\,(\text{Shr}_{\lambda/L}(\widehat{\mathbf{w}}^{[k]})))$, implies that the flag matrix remain unchanged for all $k > K$.

Similarly, for FISTA, by Lemma 3.2.4, the strict complementarity is equivalent to $\widetilde{w}_i^* \neq \pm\lambda/L$. We can let $C_2 = \min\{|\widetilde{w}_i^* - \lambda/L| - \epsilon_2, |\widetilde{w}_i^* + \lambda/L| - \epsilon_2\} > 0$ for a positive constant ϵ_2 sufficiently small to make $C_2 > 0$. We then define the following ball around

the optimal eigenvector $\widetilde{\mathbf{w}}^*_{aug}$:

$$\mathcal{B}_2 = \{\widehat{\mathbf{w}}_{aug} : \|\widetilde{\mathbf{w}}_{aug} - \widetilde{\mathbf{w}}^*_{aug}\|_\infty \leq C_2\}. \tag{3.4.20}$$

By Theorem 3.4.1 and uniqueness of the solution, the iterates $\widetilde{\mathbf{w}}^{[k]}_{aug}$ converge to $\widetilde{\mathbf{w}}^*_{aug}$. Therefore, there exists a pass K_2 such that $\widetilde{\mathbf{w}}^{[k]}_{aug}$ lies in $\mathcal{B}_2$ for all $k > K_2$. Combined with the definition of flag matrix $\widetilde{D}^{[k]} = \textsc{Diag}(\text{sign}\,(\text{Shr}_{\lambda/L}(\widetilde{\mathbf{w}}^{[k]})))$, this implies that the flag matrix remain unchanged for all $k > K_2$. □

3.4.2 Local Linear Convergence of ISTA

Once the optimal flag matrix is identified at step K_1, the iteration matrices $R^{[k]}$ and $\mathbf{R}^{[k]}_{aug}$ remain fixed for all $k > K_1$. We denote them as R^* and $\mathbf{R}^*_{aug}$ in this section.

Theorem 3.4.3 *Suppose the LASSO problem* (2.1.3) *has a unique solution and this solution satisfies the strict complementarity. Then eventually the ISTA iteration reaches a stage where it converges linearly to that unique solution.*

Proof. As shown in the proof of Lemma 3.4.2, there exists a pass number K_1 such that $\widehat{\mathbf{w}}^{[k]}_{aug}$ lies in $\mathcal{B}_1$ and $D^{[k]} = D^{[k+1]}$, for all $k > K_1$. Hence for all $k > K_1$, $R^{[k]} = R^*$ and $\mathbf{h}^{[k]} = \mathbf{h}^*$ remain invariant. By Lemma 3.2.1, the unique solution, if it exists, must correspond to a unique eigenvector of eigenvalue 1 for the matrix $\mathbf{R}^*_{aug}$. Additionally, by Lemma 3.3.1, the matrix R^* has no eigenvalue equal to 1, and by Lemma 3.2.3, all the eigenvalues must be strictly less than 1 in the absolute value. Hence, starting at the K_1-th pass, the ISTA iteration reduces to the power method on the constant matrix $\mathbf{R}^*_{aug}$ associated with the optimal flag matrix.

Let the error vector at the k-th pass of the power method be

$$\widehat{\mathbf{e}}_{aug}^{[k]} = \widehat{\mathbf{w}}_{aug}^{[k]} - \widehat{\mathbf{w}}_{aug}^{*} = \begin{pmatrix} \widehat{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix} - \begin{pmatrix} \widehat{\mathbf{w}}^{*} \\ 1 \end{pmatrix} = \begin{pmatrix} \widehat{\mathbf{e}}^{[k]} \\ 0 \end{pmatrix}.$$

Then the power iteration on $\widehat{\mathbf{w}}_{aug}^{[k]}$ yields

$$\begin{aligned} \widehat{\mathbf{w}}_{aug}^{[k+1]} &= \widehat{\mathbf{w}}_{aug}^{*} + \widehat{\mathbf{e}}_{aug}^{[k+1]} = \begin{pmatrix} R^{*}\widehat{\mathbf{w}}^{[k]} + \mathbf{h}^{*} \\ 1 \end{pmatrix} = \begin{pmatrix} R^{*}(\widehat{\mathbf{w}}^{*} + \widehat{\mathbf{e}}^{[k]}) + \mathbf{h}^{*} \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} \widehat{\mathbf{w}}^{*} + R^{*}\widehat{\mathbf{e}}^{[k]} \\ 1 \end{pmatrix} = \widehat{\mathbf{w}}_{aug}^{*} + \mathbf{R}_{aug}^{*}\widehat{\mathbf{e}}_{aug}^{[k]} \end{aligned}$$

with $\widehat{\mathbf{e}}^{[k+1]} = R^{*} \cdot \widehat{\mathbf{e}}^{[k]}$. According to Theorem 3.2.2, for any $\epsilon > 0$, there exists a matrix norm $\|\cdot\|_{\widehat{G}}$ such that $\sigma(R^{*}) < \|R^{*}\|_{\widehat{G}} \leq \sigma(R^{*}) + \epsilon < 1$. Let $\widehat{G}_{aug} = \begin{pmatrix} \widehat{G} & 0 \\ 0 & 1 \end{pmatrix}$, then

$$\|\widehat{\mathbf{e}}_{aug}^{[k+1]}\|_{\widehat{G}_{aug}} = \|\widehat{\mathbf{e}}^{[k+1]}\|_{\widehat{G}} \leq \|R^{*}\|_{\widehat{G}}\|\widehat{\mathbf{e}}^{[k+1]}\|_{\widehat{G}} = \|R^{*}\|_{\widehat{G}}\|\widehat{\mathbf{e}}_{aug}^{[k]}\|_{\widehat{G}_{aug}}.$$

Hence starting from K_1-th pass,

$$\|\widehat{\mathbf{e}}_{aug}^{[k]}\|_{\widehat{G}_{aug}} \leq O(\|R^{*}\|_{\widehat{G}}^{k-K_1}) < O((\sigma(R^{*}) + \epsilon)^{k-K_1}) \longrightarrow 0$$

as $k \longrightarrow \infty$. Therefore, $\|\widehat{\mathbf{w}}_{aug}^{[k]} - \widehat{\mathbf{w}}_{aug}^{*}\|_{\widehat{G}_{aug}}$ converges at a linear rate bounded by $\sigma(R^{*}) + \epsilon < 1$ for any $\epsilon > 0$. □

Theorem 3.4.3 indicates that when the iterate of ISTA is close enough to the optimal solution, then it converges linearly. We remark here that [13, 78] also present the same asymptotic local linear convergence result for ISTA. [13] requires the condition called strict sparsity pattern, which is identical to our strict complementarity condition while [78]

requires partial smoothness property in a more general setting.

Different from [13, 78], our analysis only focuses on finite Euclidean space so that we can take advantage of well established matrix properties. Apart from the local linear convergence results, our analysis can characterizes the convergence behavior in terms of regimes. We show how ISTA stagnates before convergence with examples in Section 3.6. Moreover, the spectral analysis we established can be applied to the FISTA's local linear convergence, which was not studied in [13, 78]. In Section 3.5, we shall make comparison between ISTA and FISTA from the perspective of our spectral analysis.

3.4.3 Local Linear Convergence of FISTA

Once the optimal flag matrix is identified, for all $k > K_2$, $\widetilde{R}^{[k]}$, $\widetilde{\mathbf{h}}^{[k]}$ and $\widetilde{\mathbf{h}}^{[k]}_{aug}$ remain fixed. We denote them as $\widetilde{R}^*$, $\widetilde{\mathbf{h}}^*$ and $\widetilde{\mathbf{h}}^*_{aug}$ in this section. Though $\mathbf{N}^{[k]}_{aug}$ still changes at each step k, one can decompose it as follows.

Lemma 3.4.4 *Assume that the LASSO problem* (2.1.3) *has a unique solution and this solution satisfies the strict complementarity. For all $k > K_2$, with K_2 defined in Lemma 3.4.2, we denote matrix $\mathbf{N}^{[\infty]}_{aug}$ as $\mathbf{N}^{[k]}_{aug}$ in which $\tau = 1$. By (3.2.13) one can write $\mathbf{N}^{[k]}_{aug} = \mathbf{N}^{[\infty]}_{aug} + (1-\tau^{[k]})\mathbf{\Delta N}_{aug}$, where*

$$\begin{aligned} \mathbf{N}^{[\infty]}_{aug} &= \begin{pmatrix} N^{[\infty]} & \widetilde{\mathbf{h}}^*_{aug} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 2\widetilde{R}^* & -\widetilde{R}^* & \widetilde{\mathbf{h}}^* \\ I & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \text{and} \quad \mathbf{\Delta N}_{aug} &= \begin{pmatrix} \Delta N & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} -\widetilde{R}^* & \widetilde{R}^* & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \end{aligned} \tag{3.4.21}$$

Consequently, $N^{[k]} = N^{[\infty]} + (1-\tau^{[k]})\Delta N$. Also, $\mathbf{N}^{[\infty]}_{aug}$ must also have a simple eigenvalue

equal to 1 *and the spectral radius of* $N^{[\infty]}$ *is strictly less than* 1. *In addition, the left and right eigenvectors of* $\mathbf{N}_{aug}^{[k]}$ *corresponding to eigenvalue 1 are the same as that of* $\mathbf{N}_{aug}^{[\infty]}$.

Proof. By Lemma 3.2.5(b), eigenvalue of $\mathbf{N}_{aug}^{[\infty]}$ must lie in the disk $\mathcal{D}(1/2, 1/2)$. Having shown that $\mathbf{N}_{aug}^{[k]}$ has only a simple eigenvalue equal to 1, by Lemma 3.2.5(c), $\widetilde{R}^*$ should have no eigenvalue equal to 1. This indicates matrix $\begin{pmatrix} 2\widetilde{R}^* & -\widetilde{R}^* \\ I & 0 \end{pmatrix}$ has no eigenvalue equal to 1. And hence $\mathbf{N}_{aug}^{[\infty]}$ must also have a simple eigenvalue equal to 1. Consequently, the absolute value of any eigenvalue of $N^{[\infty]}$ is less than 1. A simple calculation shows that the left and right eigenvectors of $\mathbf{N}_{aug}^{[k]}$ are exactly $(0, \ldots, 0, 1)$ and $\widetilde{\mathbf{w}}_{aug}^*$ ((3.4.18)), the same as that of $\mathbf{N}_{aug}^{[\infty]}$. □

Now we present one of the main results of our thesis, the local linear convergence of FISTA, as below.

Theorem 3.4.5 *Suppose the LASSO problem* (2.1.3) *has a unique solution and this solution has strict complementarity. Then eventually the FISTA iteration reaches a stage where it converges linearly to that unique solution.*

Proof. As shown in the proof of Lemma 3.4.2, there exists a pass number K_2 such that $\widetilde{\mathbf{w}}_{aug}^{[k]}$ lies in $\mathcal{B}_2$ for all $k > K_2$. Since the optimal flag matrix is identified, by Lemma 3.2.4, the unique solution, if it exists, must correspond to a unique eigenvector of eigenvalue 1 for the matrix $\mathbf{N}_{aug}^{[k]}$. Starting from K_2-th pass,

$$\begin{aligned} \widetilde{\mathbf{w}}_{aug}^{[k+1]} &= \mathbf{N}_{aug}^{[k]}\mathbf{N}_{aug}^{[k-1]}\cdots\mathbf{N}_{aug}^{[K_2]}\widetilde{\mathbf{w}}_{aug}^{[K_2]} \\ &= \begin{pmatrix} N^{[k]} & \widetilde{\mathbf{h}}_{aug}^* \\ 0 & 1 \end{pmatrix}\begin{pmatrix} N^{[k-1]} & \widetilde{\mathbf{h}}_{aug}^* \\ 0 & 1 \end{pmatrix}\cdots\begin{pmatrix} N^{[K_2]} & \widetilde{\mathbf{h}}_{aug}^* \\ 0 & 1 \end{pmatrix}\widetilde{\mathbf{w}}_{aug}^{[K_2]} \\ &= \begin{pmatrix} N^{[k]}N^{[k-1]}\cdots N^{[K_2]} & \boxed{*} \\ 0 & 1 \end{pmatrix}\widetilde{\mathbf{w}}_{aug}^{[K_2]}, \end{aligned} \tag{3.4.22}$$

where $\boxed{*}$ denotes entries that do not need to be specified in detail. For each $N^{[k]}$, we can write $N^{[k]} = N^{[\infty]} + (1-\tau^{[k]})\Delta N$. Due to the fixed flag matrix after K_2-th pass, $N^{[\infty]}$ and ΔN remain fixed. By Theorem 3.2.2, $\forall \epsilon > 0$ with $\epsilon < 1 - \sigma(N^{[\infty]})$, $\exists \, \|\cdot\|_{\widetilde{G}}$ such that $\|N^{[\infty]}\|_{\widetilde{G}} < \sigma(N^{[\infty]}) + \frac{1}{2}\epsilon < 1 - \frac{1}{2}\epsilon$. Since $\tau^{[k]} \to 1$, there must exist a pass $K_3 (> K_2)$ such that $(1-\tau^{[k]}) \cdot \|\Delta N\|_{\widetilde{G}} < \frac{1}{2}\epsilon$ for all $k > K_3$. Therefore, starting at K_3-th pass, one has $\|N^{[k]}\|_{\widetilde{G}} \leq \|N^{\infty}\|_{\widetilde{G}} + (1-\tau^{[k]})\|\Delta N\|_{\widetilde{G}} < \sigma(N^{[\infty]}) + \epsilon < 1$. As in the proof of Thm 3.4.3, let the error vector at the k-th pass for FISTA be

$$\widetilde{\mathbf{e}}^{[k]}_{aug} = \widetilde{\mathbf{w}}^{[k]}_{aug} - \widetilde{\mathbf{w}}^{*}_{aug} = \begin{pmatrix} \widetilde{\mathbf{e}}^{[k]} \\ \widetilde{\mathbf{e}}^{[k-1]} \\ 0 \end{pmatrix} \text{ where } \widetilde{\mathbf{e}}^{[k]} = \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} - \begin{pmatrix} \widetilde{\mathbf{w}}^{*} \\ \widetilde{\mathbf{w}}^{*} \\ 1 \end{pmatrix}.$$

and then

$$\widetilde{\mathbf{w}}^{[k+1]}_{aug} = \widetilde{\mathbf{w}}^{*}_{aug} + \widetilde{\mathbf{e}}^{[k+1]}_{aug} = \mathbf{N}^{[k]}_{aug}(\widetilde{\mathbf{w}}^{*}_{aug} + \widetilde{\mathbf{e}}^{[k]}_{aug}) = \widetilde{\mathbf{w}}^{*}_{aug} + \begin{pmatrix} N^{[k]} & 0 \\ 0 & 1 \end{pmatrix} \widetilde{\mathbf{e}}^{[k]}_{aug} \tag{3.4.23}$$

with $\widetilde{\mathbf{e}}^{[k+1]} = N^{[k]}\widetilde{\mathbf{e}}^{[k]}$. Let $\widetilde{G}_{aug} = \begin{pmatrix} \widetilde{G} & 0 \\ 0 & 1 \end{pmatrix}$, then

$$\|\widetilde{\mathbf{e}}^{[k+1]}_{aug}\|_{\widetilde{G}_{aug}} = \|\widetilde{\mathbf{e}}^{[k+1]}\|_{\widetilde{G}} \leq \|N^{[k]}\|_{\widetilde{G}}\|\widetilde{\mathbf{e}}^{[k]}\|_{\widetilde{G}} = \|N^{[k]}\|_{\widetilde{G}}\|\widetilde{\mathbf{e}}^{[k]}_{aug}\|_{\widetilde{G}_{aug}}.$$

Hence starting from K_3-th pass,

$$\|\widetilde{\mathbf{e}}^{[k]}_{aug}\|_{\widetilde{G}_{aug}} \leq O(\|N^{[k-1]}\|_{\widetilde{G}}\|N^{[k-2]}\|_{\widetilde{G}} \cdots \|N^{[K_3]}\|_{\widetilde{G}}) \leq O((\sigma(N^{[\infty]}) + \epsilon)^{k-K_3}) \longrightarrow 0 \tag{3.4.24}$$

as $k \longrightarrow \infty$. Therefore, $\|\widetilde{\mathbf{w}}^{[k]}_{aug} - \widetilde{\mathbf{w}}^{*}_{aug}\|_{\widetilde{G}_{aug}}$ converges at a linear rate bounded by $\sigma(N^{[\infty]}) + \epsilon < 1$ for any $\epsilon > 0$. From the proof, one can also observe that the linear rate

at step k depends on $N^{[k]}$. □

3.4.4 On the Lipschitz Constant

The previous analysis is based on $L \geq \|A^{\mathrm{T}}A\|_2$. However, this assumption is not necessary to get our main results for ISTA. [13] proves that the ISTA iterates can converge to the optimal point as long as $L > 1/2\|A^{\mathrm{T}}A\|_2$. It can be verified that our analysis allows the same L choice for ISTA.

Theorem 3.4.6 *Lemmas 3.2.1, 3.2.3, 3.3.2 and Theorem 3.4.3 hold for* $L > 1/2\|A^{\mathrm{T}}A\|_2$.

However, from our analysis, we can show that there is no guarantee that FISTA would converge if $L < \|A^{\mathrm{T}}A\|_2$. In Lemma 3.2.5(b), (3.2.15) cannot hold if $L < \|A^{\mathrm{T}}A\|_2$, indicating that some of the eigenvalues may be outside the disk $\mathcal{D}(1/2, 1/2)$. Hence convergence is not guaranteed.

3.5 Comparison and Acceleration

It is known that FISTA exhibits a global convergence rate of $O(1/k^2)$, which accelerates ISTA's $O(1/k)$ convergence rate. Compared to this worst case convergence result, we analyze how FISTA and ISTA behave through all iterations on the perspective of spectral analysis we establish in this chapter. First, we characterize one important property based on three possible regimes.

Lemma 3.5.1 *Suppose R and N have the same the flag matrix, ISTA and FISTA have the following relations:*

(a). *If FISTA is in regime [A] or [C], then so is ISTA, and vice versa.*

(b). *If FISTA is in regime [B], then so is ISTA, and vice versa.*

Proof. We note that if FISTA and ISTA start at the same iterate, we have $\widehat{D} = \widetilde{D}$, hence $\widetilde{R}$ defined in (3.2.11) is exactly operator R defined in (3.2.7).

(a). If FISTA is in regime [A] or [C], then N either has no eigenvalue equal to 1 or has a complete set of eigenvectors associated with eigenvalue 1. In other words, the augmented matrix $\mathbf{N}_{aug}$ must have a complete set of eigenvectors for eigenvalue 1. Let $\begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ 1 \end{pmatrix}$ be the eigenvector for eigenvalue 1, then

$$\begin{aligned} (N-I)\begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ 1 \end{pmatrix} &= \begin{pmatrix} (1+\tau)R & -\tau R & \mathbf{h} \\ I & -I & 0 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ 1 \end{pmatrix} = 0 \\ \iff \quad & \mathbf{w}_1 = \mathbf{w}_2 \quad \text{(by second row)} \\ \iff \quad & R\mathbf{w}_1 - \mathbf{w}_1 + \mathbf{h} = (R-I)\mathbf{w}_1 + \mathbf{h} = 0 \\ \iff \quad & \begin{pmatrix} R & \mathbf{h} \\ 0 & 1 \end{pmatrix}\begin{pmatrix} \mathbf{w}_1 \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{w}_1 \\ 1 \end{pmatrix}. \end{aligned} \tag{3.5.25}$$

Therefore, $\begin{pmatrix} \mathbf{w}_1 \\ 1 \end{pmatrix}$ becomes the eigenvector for eigenvalue 1 of $\mathbf{R}_{aug}$. R must either have no eigenvalue equal to 1 (in regime [A]) or have a complete set of eigenvectors associated with eigenvalue 1 (in regime [C]). The opposite direction follows by similar argument.

(b). Since one of the regimes [A], [B], [C] must occur, this statement can be considered as the contraposition of (a). □

This lemma suggests that both ISTA and FISTA are in the same regime as long as both operators have the same flag matrix. It motivates one to compare in each regime between FISTA and ISTA when starting from the same starting point (which results in

the same flag matrix). By assuming the same starting point and a fixed flag matrix, we have $\widehat{D}^{[k]} = \widetilde{D}^{[k]} = \widehat{D}^{[k+1]} = \widetilde{D}^{[k+1]}$ and thus $\widetilde{R} = R$, $\mathbf{h} = \widetilde{\mathbf{h}}$. We will use these notations interchangeably and omit $^{[k]}$ for anything but iterates in the following analysis. It turns out that FISTA is faster in regime [B], but not always faster in regimes [A] and [C] depending on the parameter $\tau^{[k]}$.

In Regime [B]

In regime [B], as mentioned in Section 3.3.2, there exist Jordan chains such that the difference between consecutive iterates will converge to a constant step. Let $\begin{pmatrix} \widehat{\mathbf{w}}^{[k+1]} \\ 1 \end{pmatrix}$, $\begin{pmatrix} \widehat{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix}$ and $\begin{pmatrix} \widetilde{\mathbf{w}}^{[k+1]} \\ \widetilde{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix}$, $\begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix}$ be two consecutive iterates for ISTA and FISTA, respectively. In the following lemmas, we will show that the constant step for FISTA is larger than ISTA when starting at the same point, which yields a speedup.

Lemma 3.5.2 *The constant step vector for ISTA is* $\begin{pmatrix} \mathbf{v} \\ 0 \end{pmatrix}$*, where* $\mathbf{v} = R\mathbf{v}$ *is an eigenvector of* R*.*

Proof. For ISTA, there must be a Jordan block $\mathbf{J}_R^1$ for the augmented matrix $\mathbf{R}_{aug}$. Then there exists a Jordan chain such that

$$\begin{pmatrix} R & \mathbf{h} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{w}} \\ 1 \end{pmatrix} = \begin{pmatrix} \widehat{\mathbf{w}} + \mathbf{v} \\ 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} R & \mathbf{h} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ 0 \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ 0 \end{pmatrix}. \tag{3.5.26}$$

In other words, each pass of ISTA will add a constant vector $\begin{pmatrix} \mathbf{v} \\ 0 \end{pmatrix}$ in regime [B]. □

Lemma 3.5.3 *The constant step vector for FISTA has the form* $\begin{pmatrix} c\mathbf{v} \\ c\mathbf{v} \\ 0 \end{pmatrix}$*, where* $\mathbf{v}$ *is the same* $\mathbf{v}$ *in Lemma 3.5.2,* c *is a scalar to be determined.*

Proof. Assume the constant vector is $\begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ 0 \end{pmatrix}$. Then the basic iteration of FISTA is

$$\begin{pmatrix} \widetilde{\mathbf{w}}^{[k+1]} \\ \widetilde{\mathbf{w}}^{[k]} \\ 1 \end{pmatrix} = N \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} = \begin{pmatrix} (1+\tau)R & -\tau R & \mathbf{h} \\ I & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} = \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} + \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ 0 \end{pmatrix}.$$

Due to the presence of Jordan block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, there exists a Jordan chain

$$\text{(a)}(N-I) \begin{pmatrix} \widetilde{\mathbf{w}}^{[k]} \\ \widetilde{\mathbf{w}}^{[k-1]} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ 0 \end{pmatrix} \quad \text{and} \quad \text{(b)}N \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ 0 \end{pmatrix}. \tag{3.5.27}$$

In (3.5.27a), the second row implies $\mathbf{v}_1 = \mathbf{v}_2$. Then, the first row implies $R\mathbf{v}_1 = \mathbf{v}_1$. Since both $\mathbf{v}_1$ and $\mathbf{v}$ are eigenvectors for eigenvalue 1 of R, we can write $\mathbf{v}_1 = c\mathbf{v}$ where c is a scalar to be determined. Hence the constant step should be $\begin{pmatrix} c\mathbf{v} \\ c\mathbf{v} \\ 0 \end{pmatrix}$. □

Lemma 3.5.4 *Suppose ISTA and FISTA start from the same point in the same regime [B], i.e.* $\widehat{\mathbf{w}}^{[k]} = \widetilde{\mathbf{w}}^{[k]}$*, then* c *in Lemma 3.5.3 equals* $\frac{1}{1-\tau}$*, where* τ *is a scalar close to* 1.

The constant step vector for FISTA is $\frac{1}{1-\tau}\begin{pmatrix}\mathbf{v}\\ \mathbf{v}\\ 0\end{pmatrix}$*, which is larger than* $\begin{pmatrix}\mathbf{v}\\ \mathbf{v}\\ 0\end{pmatrix}$*, the ISTA constant step.*

Proof. By Lemma 3.5.3, the equation (3.5.27) expands to

$$\begin{aligned}(N-I)\begin{pmatrix}\widetilde{\mathbf{w}}^{[k]}\\ \widetilde{\mathbf{w}}^{[k-1]}\\ 1\end{pmatrix} &= \left[\begin{pmatrix}(1+\tau)R & -\tau R & \mathbf{h}\\ I & 0 & 0\\ 0 & 0 & 1\end{pmatrix} - I\right]\begin{pmatrix}\widetilde{\mathbf{w}}^{[k]}\\ \widetilde{\mathbf{w}}^{[k-1]}\\ 1\end{pmatrix}\\ &= \begin{pmatrix}((1+\tau)R-I)\widetilde{\mathbf{w}}^{[k]} - \tau R\widetilde{\mathbf{w}}^{[k-1]} + \mathbf{h}\\ \widetilde{\mathbf{w}}^{[k]} - \widetilde{\mathbf{w}}^{[k-1]}\\ 0\end{pmatrix}\end{aligned} \tag{3.5.28}$$

which is supposed to be equal to $\begin{pmatrix}c\mathbf{v}\\ c\mathbf{v}\\ 0\end{pmatrix}$. From the second row, $\widetilde{\mathbf{w}}^{[k]} - \widetilde{\mathbf{w}}^{[k-1]} = c\mathbf{v}$ or $\widetilde{\mathbf{w}}^{[k-1]} = \widetilde{\mathbf{w}}^{[k]} - c\mathbf{v}$. Hence, the first row should be $c\mathbf{v} = ((1+\tau)R-I)\widetilde{\mathbf{w}}^{[k]} - \tau R\widetilde{\mathbf{w}}^{[k-1]} + \mathbf{h} = ((1+\tau)R-I)\widetilde{\mathbf{w}}^{[k]} - \tau R(\widetilde{\mathbf{w}}^{[k]} - c\mathbf{v}) + \mathbf{h} = (R-I)\widetilde{\mathbf{w}}^{[k]} + \mathbf{h} + c\tau\mathbf{v}$. The last equality follows from $R\mathbf{v} = \mathbf{v}$. If FISTA and ISTA start from the same point $\widehat{\mathbf{w}}^{[k]} = \widetilde{\mathbf{w}}^{[k]}$, then $c\mathbf{v} = (R-I)\widetilde{\mathbf{w}}^{[k]} + \mathbf{h} + c\tau\mathbf{v} = (R-I)\widehat{\mathbf{w}}^{[k]} + \mathbf{h} + c\tau\mathbf{v} = \mathbf{v} + c\tau\mathbf{v}$, leading to $c(1-\tau) = 1$. Hence $c = \frac{1}{1-\tau}$. □

Lemma 3.5.4 indicates that if FISTA and ISTA start from the same starting point in one specific regime [B], then it will cost FISTA fewer iterations to leave this regime with larger constant step. Hence FISTA represents an acceleration compared to ISTA in regime [B].

In Regimes [A] and [C]

On the other hand, in regimes [A] and [C], the convergence rate of the two algorithms are related to the spectral radius of R and N. Particularly, the rate of FISTA depends on τ and the iteration number, since τ is a determined sequence based on iteration numbers. Let β, γ denote an eigenvalue of R, N, respectively, and $\beta_{\max}$, $\gamma_{\max}$ denote the corresponding eigenvalues of largest absolute value. As stated in Section 3.3.2, we must have $1 > \beta_{\max}, \gamma_{\max} \geq 0$ in regimes [A] or [C]. In addition, by Lemma 3.2.5, β and γ satisfy the relation $\gamma^2 - \gamma(1+\tau)\beta + \tau\beta = 0$. Let γ_1 and γ_2 be two roots of γ. We conclude our result in the following proposition.

Proposition 3.5.5 *Suppose ISTA and FISTA start from the same point in a certain regime [A] or [C] and $D^{[k]} = D^{[k+1]}$, FISTA is faster than ISTA if $0 < \tau < \beta_{\max} < 1$ but slower if $0 < \beta_{\max} < \tau < 1$. Noting that $\beta_{\max}$ is a fixed value for one specific regime, if it is well separated from 1, with the τ growing to 1 such that $\beta_{\max} < \tau$, ISTA will be faster than FISTA toward the end.*

Proof. The roots γ of (3.2.14) are real if $\frac{4\tau}{(1+\tau)^2} < \beta$ and are complex if $\beta < \frac{4\tau}{(1+\tau)^2}$. Noting that $\tau \leq \frac{4\tau}{(1+\tau)^2} \leq 1$ with equality only if $\tau = 1$, we consider two cases.

(i). If $\frac{4\tau}{(1+\tau)^2} < \beta$, then $\tau < \beta$. Without loss of generality, $\gamma_1 = \max\{\gamma_1, \gamma_2\} = \frac{(1+\tau)\beta}{2} + \sqrt{\frac{(1+\tau)^2\beta^2 - 4\tau\beta}{4}} < \frac{(1+\tau)\beta}{2} + \sqrt{\frac{(1+\tau)^2\beta^2 - 4\beta^2}{4}} < \beta$. The first inequality is due to $\beta > \tau$ and the second one is due to $\tau < 1$.

(ii). If $\beta < \frac{4\tau}{(1+\tau)^2}$, then γ_1 and γ_2 are a conjugate complex pair such that $|\gamma_1|^2 = \gamma_1\gamma_2 = \tau\beta$. If $\tau < \beta < \frac{4\tau}{(1+\tau)^2}$, then $|\gamma_1| = \sqrt{\tau\beta} < \beta$. If $\beta < \tau < \frac{4\tau}{(1+\tau)^2}$, then $|\gamma_1| = \sqrt{\tau\beta} > \beta$.

If FISTA were to continue long enough, eventually $\tau^{[k]}$ will become larger than $\beta_{\max} < 1$ at some step K_4, at which point the asymptotic convergence rate for FISTA will be slower than that for ISTA. □

Proposition 3.5.5 concludes that if the starting points are the same in regimes [A]

or [C], then ISTA will first be slower but then be faster as the iteration progresses. Therefore, it is advantageous to make FISTA iteration switch to ISTA once it reaches the final regime. We implement this idea in numerical examples and name it hybrid F/ISTA.

A Heuristic Algorithm

The above analysis would indicate that one should try to take advantage of the generally faster $O(1/k^2)$ rate of convergence of FISTA, but switch to ISTA when it becomes faster during the final regime. We test this idea using a Hybrid F/ISTA method in which we run FISTA until reaching the final linear regime and then switch to ISTA. The result is illustrated in the examples. However, there is no way in practice to know when one reaches the final regime without knowing the optimal solution. So we also propose a simple heuristic algorithm in which both ISTA and FISTA iterates are computed at every iteration, choosing whichever iterate shows greater progress measured in terms of the decrease in the objective function (2.1.3). This heuristic algorithm is given as Alg. 5, with initialization $\mathbf{y}^{[1]} = \widetilde{\mathbf{x}}^{[0]} \in \mathbb{R}^n$ and $t^{[0]} = t^{[1]} = 1$.

Algorithm 11 One pass of the heuristic algorithm

Input: Start with $t^{[k]}$, $t^{[k-1]}$, $\mathbf{x}^{[k-1]}$ and $\mathbf{x}^{[k-2]}$.

1: Set $\mathbf{y}^{[k]} = \mathbf{x}^{[k-1]} + \frac{t^{[k-1]}-1}{t^{[k]}}(\mathbf{x}^{[k-1]} - \mathbf{x}^{[k-2]})$.

2: $\widetilde{\mathbf{x}}^{[k]} = \mathrm{Shr}_{\lambda/L}((I - 1/L A^{\mathrm{T}}A)\mathbf{y}^{[k]} + 1/L A^{\mathrm{T}}\mathbf{b})$.

3: $\widehat{\mathbf{x}}^{[k]} = \mathrm{Shr}_{\lambda/L}((I - 1/L A^{\mathrm{T}}A)\mathbf{x}^{[k-1]} + 1/L A^{\mathrm{T}}\mathbf{b})$.

4: If $F(\widehat{\mathbf{x}}^{[k]}) > F(\widetilde{\mathbf{x}}^{[k]})$, then $\mathbf{x}^{[k]} = \widetilde{\mathbf{x}}^{[k]}$. Else, $\mathbf{x}^{[k]} = \widehat{\mathbf{x}}^{[k]}$.

5: Set $t^{[k+1]} = \frac{1+\sqrt{1+4t^{[k]2}}}{2}$.

Output: $t^{[k+1]}$, $t^{[k]}$, $\widetilde{\mathbf{x}}^{[k]}$, $\widetilde{\mathbf{x}}^{[k-1]}$ for next pass.

Though this updating rule is simple and lacks the theoretical $O(1/k^2)$ global convergence rate, the experimental results show that it can converge very fast, sometimes faster than we expect. The computational cost of one iteration of the heuristic algorithm is the same as for FISTA plus one extra shrinkage operation and two evaluations of the objective function. However, in our following examples, we observe that the proposed

algorithm can sometimes be so much faster than ISTA or FISTA alone that the overall cost can be much less. An alternative acceleration heuristic (with similar behavior) was discussed in [91].

3.6 Numerical Examples

Example 1**.** We illustrate the eigen-analysis of the behavior of ISTA and FISTA on a uniform randomly generated LASSO problem. Specifically, in problem (2.1.3), A and $\mathbf{b}$ are generated independently by a uniform distribution over $[-1, 1]$, A being 20×40, $\lambda = 0.1$. Since A is drawn by a continuous distribution, as noted in Lemma 4 of [118], problem (2.1.3) must have a unique solution. Figures 3.1 & 3.2 show the ISTA and FISTA convergence behavior. The figures show the error of $\mathbf{x}$, $\|\mathbf{x}^{[k]} - \mathbf{x}^*\|$ (A: top curve) and the difference between two consecutive iterates of $\mathbf{x}$: $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|$ (B: bottom curve).

Figure 3.2 (First) shows the behavior of ISTA. ISTA takes 5324 iterations to converge and the flag matrix D changes 25 times in total. During the first 174 iterations, the iteration passes through a few transitional phases and the flag matrix D changes 20 times. After that, from iteration 175 to 483, it stays in regime [B] with an invariant D. Then from iteration 484 to 530, from 531 to 756, from 767 to 4722 and from 4723 to 4972, it passes through four different regimes [B]. Within each regime [B], the flag matrix D is invariant. According to our analysis in Section 3.3.2, there exists a Jordan chain in each of these regimes [B], indicating that we are indeed in a "constant step" regime. In other words, the difference between two consecutive iterates $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|$ quickly converges to $\mathbf{R}_{aug}$'s eigenvector for eigenvalue 1 in each of these regimes [B]. Taking iterations from 767 to 4722 for example, one could notice curve **B** in Figure 3.2 (First) that $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|$ is a constant from iteration 767 to 4722. Finally, at iteration 4973, it reaches and stays in the final regime [A], converging linearly in 351 steps. Indeed, the

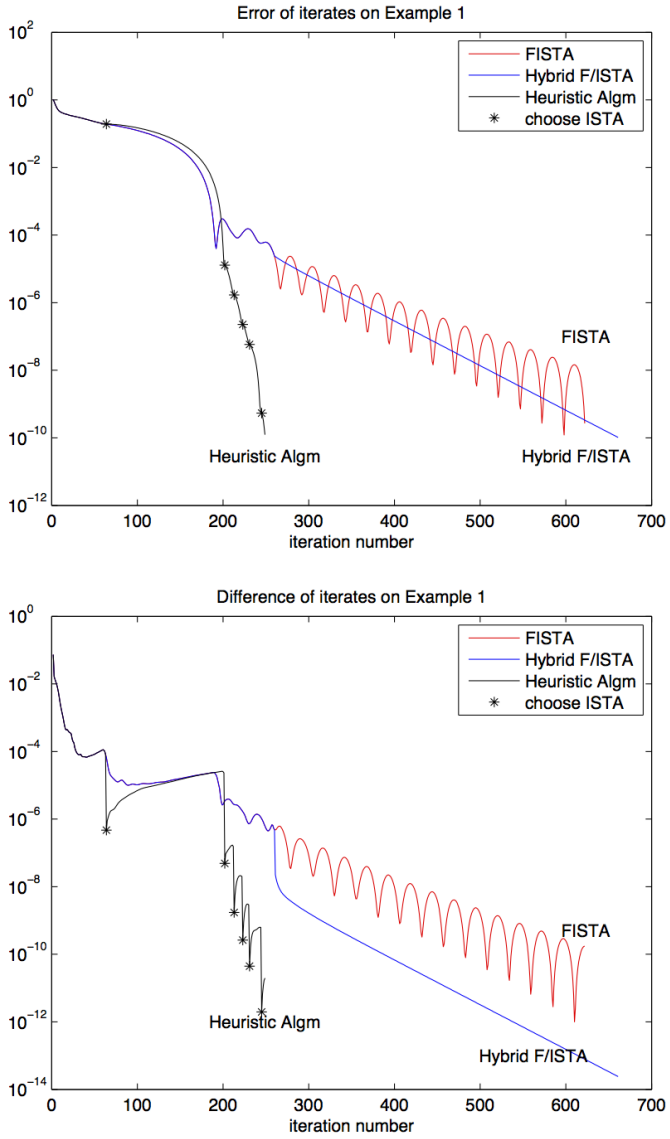


Figure 3.1: First: Error of iterates of FISTA, Hybrid F/ISTA and Heuristic Algorithm. Second: Difference of iterates of FISTA, Hybrid F/ISTA and Heuristic algorithms. The star * on the Heuristic Algm curve marks the iterations where the ISTA iterate was selected.

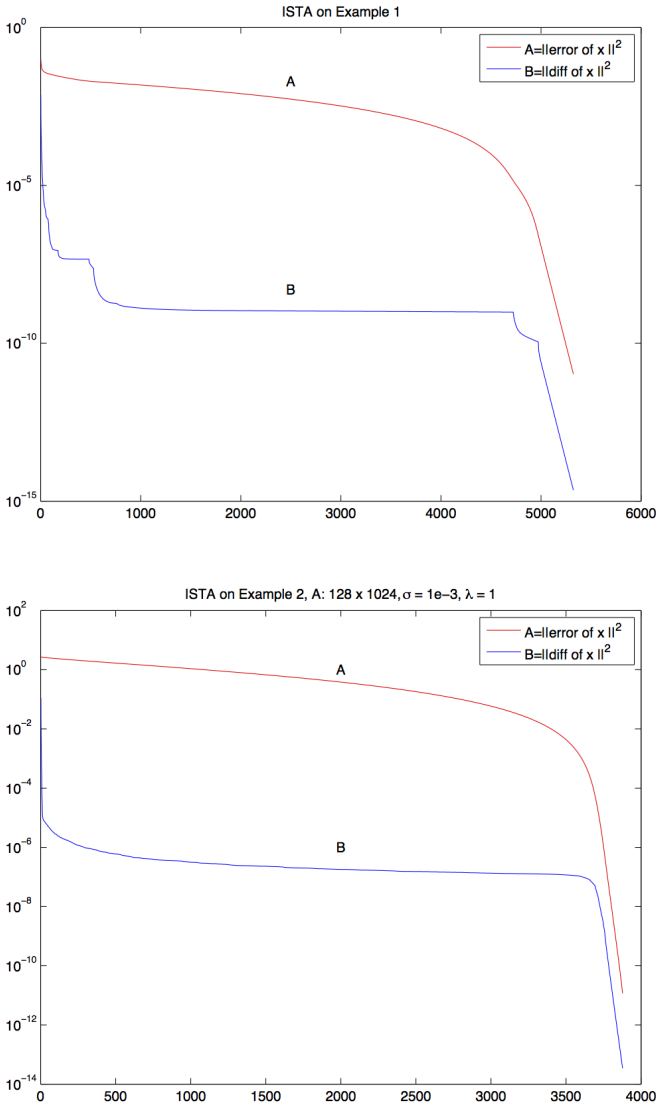


Figure 3.2: First: ISTA on Example 1. Second: Example 2: Curves **A**: $\|\mathbf{x}^{[k]} - \mathbf{x}^*\|^2$. **B**: $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|^2$.

iterates are close enough to the final optimum so that the flags never change again. The linear convergence rate depends on the spectral radius of R, i.e. upper left part of $\mathbf{R}_{aug}$, which is $\rho(R) = 0.9817$, separated from the $\mathbf{R}_{aug}$'s largest eigenvalue 1, consistent with Theorem 3.4.3.

Figure 3.1 (FISTA) shows the behavior of FISTA. FISTA takes 622 iterations to converge and the flag matrix D changes 42 times in total. After flag matrix D changes 42 times in initial 258 iterations, it reaches the final regime [A] at iteration 259 and converges linearly in 363 steps. Since $\mathbf{N}_{aug}$ varies at each iteration due to varying τ, the convergence rate changes very slightly step by step. The spectral radius of N, i.e. upper left part of the operator $\mathbf{N}_{aug}$ in the last step is $\rho(N) = 0.9914$. Actually, the largest eigenvalues of N are a complex conjugate pair, $0.9843 \pm 0.1185i$. They are complex numbers because of the increasing τ, as stated in Proposition 3.5.5 in Section 3.5. Hence based on the power method, in the final regime, the convergence to eigenvector for eigenvalue 1 of $\mathbf{N}_{aug}$ will oscillate between the conjugate complex pair. This explains why curves in Figure 3.1 (FISTA) oscillate in the latter part of the FISTA convergence.

We made three more remarks with our analysis in Section 3.5 on this example.

1. It costs FISTA many fewer steps (259 iterations) than ISTA (4973 iterations) to get to the final regime. The main reason is that FISTA has much larger constant steps in regime [B] so that it can jump out of that regime more quickly. Though this will lead to more changes of regimes (flag matrix D changes 42 times, 17 more times than ISTA), the overall iteration numbers have been cut down, consistent with Lemma 3.5.4. One can also notice this in Figure 3.1 (FISTA) that difference between iterates do not remain constant for many iterations, with the process transitioning into the final regime.

2. Figure 3.1 (Hybrid F/ISTA) shows the behavior of hybrid F/ISTA idea illustrated in Section 3.5. Particularly for this example, it runs FISTA until it reaches final regime and then switches to ISTA at iteration 260. At step 260, $\tau = 0.9886$, larger than

ISTA rate 0.9817, predicting that ISTA should converge faster than FISTA. Though Hybrid F/ISTA converges in 661 iterations, more than 622 of FISTA iterations, it doesn't contradict with our analysis. In Figure 3.1, from the gradient of FISTA and Hybrid F/ISTA curve, one could observe that Hybrid F/ISTA converges faster than the upper bound of FISTA.

3. Figure 3.1 (Heuristic Algm) shows the behavior of heuristic algorithm established in Section 3.5. It converges in only 212 iterations with the same accuracy. Though it costs extra shrinkage operations and objective value evaluations, the overall cost can still be much less.

Figure 3.3 shows the eigenvalues of the operators $\mathbf{R}_{aug}$ and $\mathbf{N}_{aug}$ during the final regime. One notices that the eigenvalues for the $\mathbf{R}_{aug}$ from (3.2.7) lie strictly on the interval $(0, 1)$ and eigenvalues for $\mathbf{N}_{aug}$ lie close to the boundary but strictly inside the circle $\mathcal{D}(1/2, 1/2)$ (except for 0 and 1), consistent with Lemmas 3.2.3 & 3.2.5.

Example 2. We consider an example of compressive sensing. The purpose of this example is to show and compare the convergence behavior of different algorithms mentioned in previous sections to support our analysis. Suppose there exists a true sparse signal represented by a n-th dimension vector $\mathbf{x}$ with k non-zero elements. We observe the image of $\mathbf{x}_s$ under the linear transformation $A\mathbf{x}_s$, where A is the so-called measurement matrix. Our observation thus should be

$$\mathbf{b} = A\mathbf{x}_s + \epsilon \tag{3.6.29}$$

where ϵ is the observation noise. The goal is to recover the sparse vector $\mathbf{x}_s$ from the measurement matrix A and observation $\mathbf{b}$. For this example, we let $A \in \mathbb{R}^{m\times n}$ be Gaussian matrix whose elements are *i.i.d.* distributed as $\mathcal{N}(0, 1)$ with $m = 128$ and $n = 1024$, ϵ be a vector whose elements are *i.i.d.* distributed as $\mathcal{N}(0, \sigma^2)$ with $\sigma = 10^{-3}$.

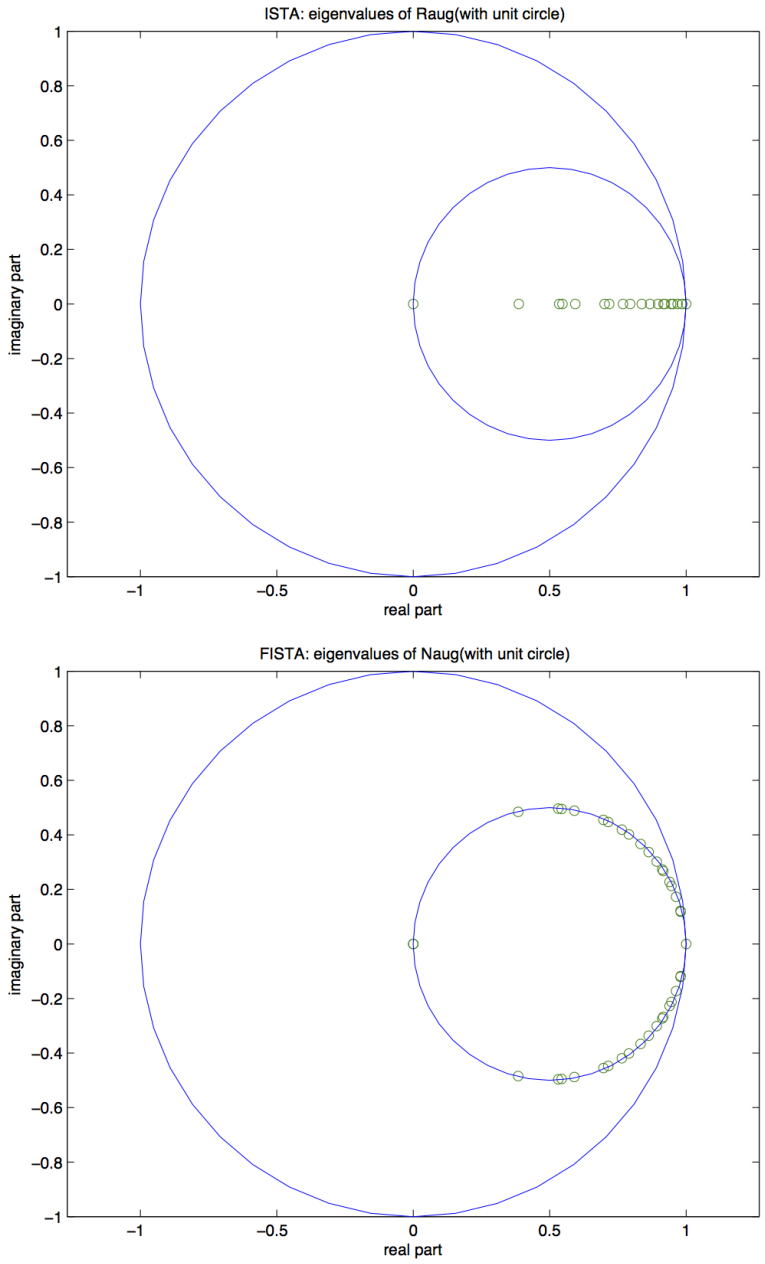


Figure 3.3: ISTA (First) and FISTA (Second) on Example 1: Eigenvalues of ISTA operator $\mathbf{R}_{aug}$ and FISTA operator $\mathbf{N}_{aug}$ on the complex plane during the last regime of the iteration process. The unit circle and $\mathcal{D}(1/2, 1/2)$ are shown for reference.

The original true signal for the problem is generated by choosing the locations of $\mathbf{x}$'s $k(=10)$ nonzeros uniformly at random, then setting those locations to values drawn from $\mathcal{N}(0, 2^2)$. We solve this compressive sensing problem by model (2.1.3) with $\lambda = 1$ and illustrate the convergence behavior of four methods: ISTA, FISTA, Hybrid F/ISTA and the heuristic algorithm. Figure 3.2 (Second) shows ISTA's convergence behavior. Figures 3.4 respectively show the error and difference of the iterates of other three methods.

ISTA: It costs 3763 iterations to reach the final regime, finally converging in altogether 3882 iterations. The linear convergence rate is the second largest eigenvalue of $\mathbf{R}_{aug}$, which equals to 0.9584, well separated from 1.

FISTA: It costs 333 iterations to reach the final regime and converges in totally 515 iterations. The linear convergence rate at pass k, as shown in Theorem 3.4.5, depends on the second largest eigenvalue of $\mathbf{N}_{aug}^{[k]}$. The linear rate is 0.9746 with $\tau = 0.9911$ at step 333 and the rate is 0.9761 with $\tau = 0.9942$ at step 515.

The iteration number for FISTA obviously is shorter than ISTA. It can be seen that in Figure 3.2 (Second, curve B) that the difference of ISTA iterates remain at a constant number for many iterations. This is because they are in the constant regimes such that the difference between consecutive iterates are converging to a constant vector. From Figure 3.4 (Second), one can observe that the difference of FISTA iterates doesn't stagnate for as many iterations as ISTA because it has a larger constant step size, as predicted in Section 3.5.

Hybrid F/ISTA: By the time FISTA reaches the final regime, $\tau = 0.9911$ which is already greater than ISTA rate 0.9584, predicting that switching to ISTA at this point would be advantageous by Proposition 3.5.5. Particularly, one runs FISTA iterates until the arrival of the final regime at step 334. Then one switches to ISTA until convergence so that a faster linear rate is obtained. The algorithm of this idea converges only in 437 iterations with the same accuracy compared to 515 iterations of FISTA. One can observe

the acceleration in Figure 3.4 (First).

Heuristic Algorithm: Finally, we test our heuristic algorithm developed in Section 3.5 on this example. Basically, at each iteration, the algorithm compares the objective value by running ISTA and FISTA, and update the iterate with the lower value. From our analysis in Section 3.5, the heuristic algorithm should mostly run FISTA before final linear regime and switch to ISTA very often toward the end. We indeed observe this phenomenon in Figure 3.4. The heuristic algorithm converges only in 218 iterations. Though it loses the theoretical global $O(1/k^2)$ rate, it has a better practical performance. This, combined with the Hybrid F/ISTA idea, is consistent with our analysis of switching iterations to ISTA towards the end.

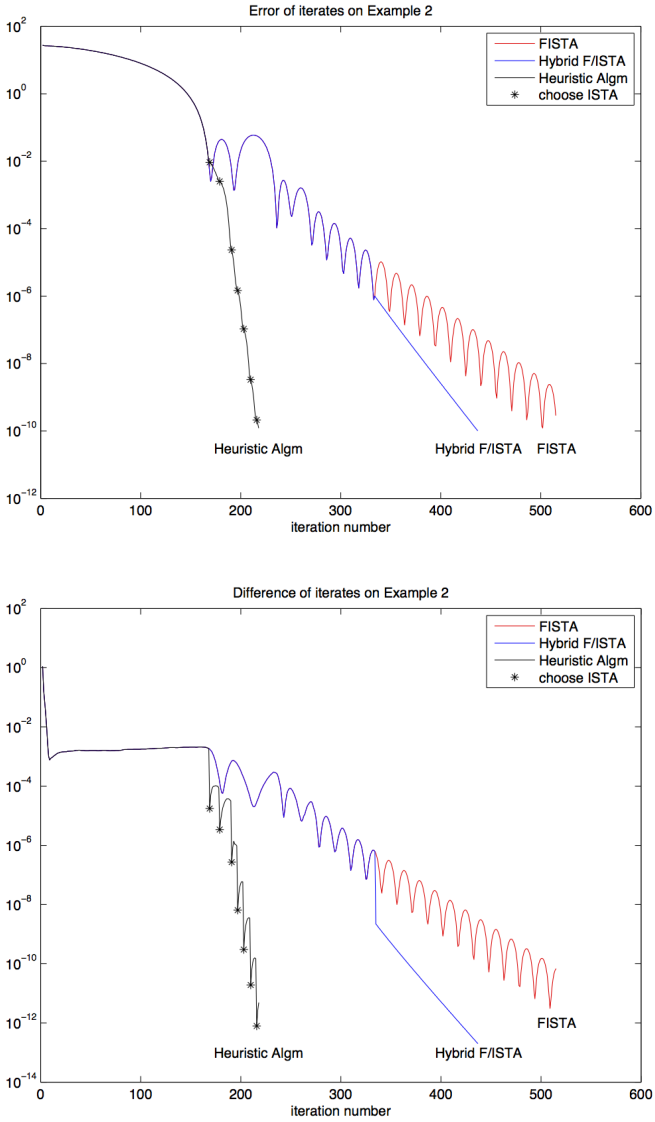


Figure 3.4: First: Error of iterates of FISTA, Hybrid F/ISTA and Heuristic Algorithm. Second: Difference of iterates of FISTA, Hybrid F/ISTA and Heuristic algorithms. The star * on the Heuristic Algm curve marks the iterations where the ISTA iterate was selected.

Chapter 4

Local Linear Convergence of ADMM and Coordinate Descent on the LASSO Model

In the last chapter, we investigate the local convergence of proximal gradient method and accelerated proximal gradient method on the LASSO model. We show both iterations can converge linearly when close enough to the optimal solution by reformulating into homogeneous matrix recurrence formulations and applying spectral analysis. In this chapter, we look into Alternating Minimization Method of Multipliers (ADMM) and Coordinate Descent (CD) on the LASSO model. We show that the same technique in the last chapter can be extended to more algorithms and local linear bound can be achieved. Moreover, we compare all methods from this chapter and last chapter together to give a exclusive summary.

The following of this chapter is organized as follows. We first introduce the basic iterations of ADMM and Coordinate Descent on the LASSO problem in Section 4.1. In Section 4.2, we extend the same technique as in the last chapter, showing that linear

convergence of ADMM is reached eventually on the LASSO problem. It is important to note that, though the LASSO problem is a special case of a quadratic programming, the specific splitting used by ADMM for the LASSO problem differs from that for a generic quadratic programming. For the LASSO problem, a unique shrinkage splitting is applied in a much more efficient way. Hence the analysis of ADMM in this chapter is independent and cannot be considered as a special case of [11]. In Section 4.3, we establish a natural relationship between the iterations of the cyclic CD and the Gauss-Seidel method when close enough to the optimal solution so that linear convergence is guaranteed eventually under certain mild conditions. Finally, through numerical examples in Section 4.4, we compare their convergence behavior with ISTA and FISTA illustrated in the last chapter.

4.1 ADMM and Coordinate Descent Iterations

We have presented a general introduction on ADMM and Coordinate Descent and their global convergence results on general problem in Section 1.3. Restricted to the local convergence behavior, [11] started to look into a quadratic programming solved by ADMM. Essentially, [11] represents the ADMM iteration in a novel way as a matrix recurrence and apply the spectral analysis. The behavior of ADMM iterations are characterized as the "regimes" and the linear convergence is showed towards the end. Later, [60] also showed the local linear convergence for a quadratic programming problem with a different approach.

Recall the LASSO problem is

$$\min_{\mathbf{x}\in\mathbb{R}^p} F(\mathbf{x}) = \tfrac{1}{2}\|A\mathbf{x}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{4.1.1}$$

Following Algorithm 4 in Section 1.3, a typical splitting for LASSO problem is to use variable $\mathbf{x}$ for the least square loss function and $\mathbf{z}$ for the ℓ_1-norm regularizer. The

modified LASSO problem is

$$\begin{aligned} \min_{\mathbf{x}} \quad & 1/2\|A\mathbf{x}-\mathbf{b}\|_2^2+\lambda\|\mathbf{z}\|_1 \\ \text{s.t.} \quad & \mathbf{x}-\mathbf{z}=0 \end{aligned} \tag{4.1.2}$$

with augmented Lagrangian function being

$$\begin{aligned} \mathcal{L}_\rho &= 1/2\|A\mathbf{x}-\mathbf{b}\|_2^2+\lambda\|\mathbf{z}\|_1+\mathbf{y}^{\mathrm{T}}(\mathbf{x}-\mathbf{z})+\rho/2\|\mathbf{x}-\mathbf{z}\|_2^2 \\ &= 1/2\|A\mathbf{x}-\mathbf{b}\|_2^2+\lambda\|\mathbf{z}\|_1+\rho/2\|\mathbf{x}-\mathbf{z}+\mathbf{u}\|_2^2-\rho/2\|\mathbf{u}\|_2^2 \end{aligned}$$

where ρ is a penalty parameter, $\mathbf{y}$ is the Lagrange multipliers for the additional constraint $\mathbf{x}-\mathbf{z}=0$, and $\mathbf{u}=\mathbf{y}/\rho$ in the second equation is the scaled dual variable. The ADMM iterative scheme for this typical splitting is then as follows,

$$\begin{cases} \mathbf{x}^{[k+1]}=\operatorname{argmin}_{\mathbf{x}} 1/2\|A\mathbf{x}-\mathbf{b}\|_2^2+\rho/2\|\mathbf{x}-\mathbf{z}^{[k]}+\mathbf{u}^{[k]}\|_2^2 \\ \mathbf{z}^{[k+1]}=\operatorname{argmin}_{\mathbf{z}} \rho/2\|\mathbf{x}^{[k+1]}-\mathbf{z}+\mathbf{u}^{[k]}\|_2^2+\lambda\|\mathbf{z}\|_1 \\ \mathbf{u}^{[k+1]}=\mathbf{u}^{[k]}+\mathbf{x}^{[k+1]}-\mathbf{z}^{[k]}. \end{cases} \tag{4.1.3}$$

Each step of (4.1.3) can be solved in closed form, leading to the ADMM iteration consisting of the following steps repeated until convergence. Here we use the notation $\mathbf{z}^{[k]}$, $\mathbf{u}^{[k]}$ and $\mathbf{z}^{[k+1]}$, $\mathbf{u}^{[k+1]}$ to denote the iterates at the beginning and the end of the k-th pass, respectively. ρ is a given fixed penalty. The shrinkage operator is defined in the

Algorithm 12 One pass of ADMM for LASSO problem

Input: Start with $\mathbf{z}^{[k]}$ and $\mathbf{u}^{[k]}$.

1: Set $\mathbf{x}^{[k+1]}=(A^{\mathrm{T}}A+\rho I)^{-1}[A^{\mathrm{T}}\mathbf{b}+\rho(\mathbf{z}^{[k]}-\mathbf{u}^{[k]})]$.

2: Set $\mathbf{z}^{[k+1]}=\mathrm{Shr}_{\lambda/\rho}(\mathbf{x}^{[k+1]}+\mathbf{u}^{[k]})$.

3: Set $\mathbf{u}^{[k+1]}=\mathrm{Thr}_{\lambda/\rho}(\mathbf{x}^{[k+1]}+\mathbf{u}^{[k]})$.

Output: $\mathbf{z}^{[k+1]}, \mathbf{u}^{[k+1]}$.

same way as (3.1.2) and thresholding operator defined below is also applied elementwise to vectors.

$$\mathrm{Thr}_\xi(s) = s - \mathrm{Shr}_\xi(s) = \begin{cases} \xi & \text{if } s \geq \xi \\ s & \text{if } -\xi < s < \xi \\ -\xi & \text{if } s \leq -\xi, \end{cases} \tag{4.1.4}$$

As for cyclic coordinate descent method, though it is widely used due to its easy update rule, there is no clear result on the local convergence behavior. [121] proved the global linear convergence under several error bound conditions and [107] showed the global $O(1/k)$ convergence rate for the LASSO problem.

Note that the regularized term in the LASSO problem is separable, i.e. $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$. The CD algorithm can take advantage of this structure to update along each coordinate cyclically. We let x_i be the i-th component of variable $\mathbf{x}$ and all components other than i are denoted as the vector $\mathbf{x}_{-i}$. Following (1.3.7), by minimizing the objective function over x_i with fixed x_j, $j \neq i$, one gets

$$A_i^{\mathrm{T}}(A\mathbf{x} - \mathbf{b}) + \lambda s_i \in \partial_i F(\mathbf{x})$$

where $s_i \in \partial |x_i|$. Taking the left hand side to be zero,

$$A_i^{\mathrm{T}} A_i x_i + A_i^{\mathrm{T}} A_{-i}\mathbf{x}_{-i} - A_i^{\mathrm{T}}\mathbf{b} + \lambda s_i = 0.$$

This leads to the x_i-update:

$$x_i = \mathrm{Shr}_\lambda(A_i\mathbf{b} - A_i^{\mathrm{T}} A_{-i}\mathbf{x}_{-i})/\|A_i^{\mathrm{T}} A_i\|.$$

Therefore, the Coordinate Descent for the LASSO problem can be summarized as follows.

Algorithm 13 One pass of Coordinate Descent for LASSO problem

Input: Start with $\mathbf{x}^{[k]}$.

1: **for** $i = 1, 2, \cdots, n$ **do**

2: $\quad x_i^{([k+1],i)} = \mathrm{Shr}_\lambda(A_i\mathbf{b} - A_i^{\mathrm{T}} A_{-i}\mathbf{x}_{-i}^{([k],i-1)})/\|A_i^{\mathrm{T}} A_i\|$.

3: $\quad \forall j \neq i,\ x_j^{([k],i)} = x_j^{([k],i-1)}$.

4: **end for**

Output: $\mathbf{x}^{[k+1]}$ for next pass.

4.2 ADMM for the LASSO Model

4.2.1 ADMM as Matrix Recurrence

Recall the ADMM iteration for the LASSO model is

$$\begin{cases} \mathbf{x}^{[k+1]} &= (A^{\mathrm{T}}A + \rho I)^{-1}[A^{\mathrm{T}}\mathbf{b} + \rho(\mathbf{z}^{[k]} - \mathbf{u}^{[k]})] \\ \mathbf{z}^{[k+1]} &= \mathrm{Shr}_{\lambda/\rho}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]}) \\ \mathbf{u}^{[k+1]} &= \mathrm{Thr}_{\lambda/\rho}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]}). \end{cases}$$

To reformulate in a similar way as in ISTA and FISTA on the LASSO problem, we use two auxiliary variables $\mathbf{w}^{[k]}$ and $\mathbf{d}^{[k]}$ to carry the iteration instead of $\mathbf{z}^{[k]}$ and $\mathbf{u}^{[k]}$. Specifically, we let, for all k, the common iterate be $\mathbf{w}^{[k]} = \mathbf{z}^{[k]} + \mathbf{u}^{[k]}$ and $\mathbf{d}^{[k]}$ be a vector defined elementwise as

$$d_i^{[k]} = \mathrm{sign}(\mathrm{Shr}_{\lambda/\rho}(w_i^{[k]})) = \begin{cases} 1 & \text{if } w_i^{[k]} > \lambda/\rho \\ 0 & \text{if } -\lambda/\rho \leq w_i^{[k]} \leq \lambda/\rho \\ -1 & \text{if } w_i^{[k]} < -\lambda/\rho. \end{cases} \tag{4.2.5}$$

Since $\mathbf{w}^{[k+1]} = \mathbf{z}^{[k+1]} + \mathbf{u}^{[k+1]} = \mathbf{x}^{[k+1]} + \mathbf{u}^{[k]}$, one can write ADMM steps of $\mathbf{z}^{[k]}$ and $\mathbf{u}^{[k]}$ in terms of $\mathbf{w}^{[k+1]}$ and the matrix $D^{[k]} = \text{diag}(\mathbf{d}^{[k+1]})$

$$\begin{cases} \mathbf{z}^{[k+1]} &= \text{Shr}_{\lambda/\rho}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]}) = \text{Shr}_{\lambda/\rho}(\mathbf{w}^{[k+1]}) = (D^{[k+1]})^2\mathbf{w}^{[k+1]} - \lambda/\rho\,\mathbf{d}^{[k+1]} \\ \mathbf{u}^{[k+1]} &= \text{Thr}_{\lambda/\rho}(\mathbf{x}^{[k+1]} + \mathbf{u}^{[k]}) = \text{Thr}_{\lambda/\rho}(\mathbf{w}^{[k+1]}) = (I - (D^{[k+1]})^2)\mathbf{w}^{[k+1]} + \lambda/\rho\,\mathbf{d}^{[k+1]}. \end{cases}$$

With the updating rule of ADMM and above two identities, one can obtain the $\mathbf{x}^{[k]}$-update in terms of $\mathbf{w}^{[k]}$ and $D^{[k]}$

$$\begin{aligned} \mathbf{x}^{[k+1]} &= (A^{\mathrm{T}}A + \rho I)^{-1}[A^{\mathrm{T}}\mathbf{b} + \rho(\mathbf{z}^{[k]} - \mathbf{u}^{[k]})] \\ &= (A^{\mathrm{T}}A + \rho I)^{-1}[A^{\mathrm{T}}\mathbf{b} - 2\lambda\mathbf{d}^{[k]} + \rho(2(D^{[k]})^2 - I)\mathbf{w}^{[k]}]. \end{aligned}$$

Hence the update formula for $\mathbf{w}^{[k+1]}$ can now be expressed explicitly as follows:

$$\begin{aligned} \mathbf{w}^{[k+1]} &= \mathbf{x}^{[k+1]} + \mathbf{u}^{[k]} \\ &= [(I - (D^{[k]})^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2(D^{[k]})^2 - I)]\mathbf{w}^{[k]} \\ &\quad + \lambda/\rho\,\mathbf{d}^{[k]} + (A^{\mathrm{T}}A + \rho I)^{-1}(A^{\mathrm{T}}\mathbf{b} - 2\lambda\mathbf{d}^{[k]}) \\ &= M^{[k]}\mathbf{w}^{[k]} + \mathbf{h}^{[k]} \end{aligned} \tag{4.2.6}$$

where we denote

$$\begin{aligned} M^{[k]} &= (I - (D^{[k]})^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2(D^{[k]})^2 - I) \\ \mathbf{h}^{[k]} &= \lambda/\rho\,\mathbf{d}^{[k]} + (A^{\mathrm{T}}A + \rho I)^{-1}(A^{\mathrm{T}}\mathbf{b} - 2\lambda\mathbf{d}^{[k]}) \end{aligned} \tag{4.2.7}$$

throughout this chapter. So the ADMM update with $\mathbf{x}^{[k+1]}$, $\mathbf{z}^{[k+1]}$, $\mathbf{u}^{[k+1]}$ can be modified to the following procedure using the new variables $\mathbf{d}^{[k+1]}$ and $\mathbf{w}^{[k+1]}$.

Algorithm 14 is mathematically equivalent to Algorithm 12 and we will use this form to characterize its convergence properties. We note that Step 1 of Algorithm 14 can be

Algorithm 14 One pass of ADMM

Input: Start with $\mathbf{w}^{[k]}$, $D^{[k]}$.

1: Set $\mathbf{w}^{[k+1]} = [(I - (D^{[k]})^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2(D^{[k]})^2 - I)]\mathbf{w}^{[k]} + {}^{\lambda}\!/_{\rho}\mathbf{d}^{[k]} + (A^{\mathrm{T}}A + \rho I)^{-1}(A^{\mathrm{T}}\mathbf{b} - 2\lambda\mathbf{d}^{[k]})$.

2: Set $D^{[k+1]} = \textsc{Diag}(\mathrm{sign}\,(\mathrm{Shr}_{{}^{\lambda}\!/_{\rho}}(\mathbf{w}^{[k+1]})))$.

Output: $\mathbf{w}^{[k+1]}$, $D^{[k+1]}$.

written as a homogeneous matrix recurrence.

$$\begin{aligned}\begin{pmatrix}\mathbf{w}^{[k+1]}\\ 1\end{pmatrix} =& \mathbf{M}_{aug}^{[k]}\begin{pmatrix}\mathbf{w}^{[k]}\\ 1\end{pmatrix} = \begin{pmatrix}M^{[k]} & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}\begin{pmatrix}\mathbf{w}^{[k]}\\ 1\end{pmatrix}\\ =& \begin{pmatrix}(I - (D^{[k]})^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2(D^{[k]})^2 - I) & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}\begin{pmatrix}\mathbf{w}^{[k]}\\ 1\end{pmatrix}\end{aligned} \tag{4.2.8}$$

where we denote $\mathbf{M}_{aug}^{[k]}$ as $\begin{pmatrix}M^{[k]} & \mathbf{h}^{[k]}\\ 0 & 1\end{pmatrix}$, the augmented matrix of $M^{[k]}$, in this section.

Analogous to Lemma 3.2.1 for ISTA and Lemma 3.2.4 for FISTA, the following lemma shows the equivalence between the fixed point of the constructed matrix recurrence (4.2.8) and the KKT point of the LASSO problem.

Lemma 4.2.1 *Suppose* $\begin{pmatrix}\mathbf{w}\\ 1\end{pmatrix}$ *is an eigenvector corresponding to eigenvalue 1 of* $\mathbf{M}_{\mathrm{aug}}$ *in* (4.2.8) *and* $D = D^{[k+1]} = D^{[k]} = \textsc{Diag}(\mathbf{d})$ *with entries* $d_i = 1$ *if* $w_i \geq {}^{\lambda}\!/_{\rho}$, $d_i = -1$ *if* $w_i \leq -{}^{\lambda}\!/_{\rho}$ *and* $d_i = 0$ *if* $-{}^{\lambda}\!/_{\rho} \leq w_i \leq {}^{\lambda}\!/_{\rho}$. *Then the primal variable* $\mathbf{x} = \mathbf{z} = Shr_{{}^{\lambda}\!/_{\rho}}(\mathbf{w})$ *and dual variable* $\boldsymbol{\nu} = {}^{\rho}\!/_{\lambda}\, Thr_{{}^{\lambda}\!/_{\rho}}(\mathbf{w})$ *satisfy 1st order KKT conditions* (2.1.4) *and* (2.1.5). *Conversely, if* $\mathbf{x}$ *and* $\boldsymbol{\nu}$ *satisfy the KKT conditions, then* $\begin{pmatrix}\mathbf{w}\\ 1\end{pmatrix}$, *with* $\mathbf{x} = \mathbf{z}$, $\mathbf{u} = {}^{\lambda}\!/_{\rho}\boldsymbol{\nu}$ *and* $\mathbf{w} = \mathbf{z} + \mathbf{u}$, *is an eigenvector of* $\mathbf{M}_{\mathrm{aug}}$ *corresponding to eigenvalue 1, where* $\mathbf{M}_{\mathrm{aug}}$ *is defined as in* (4.2.8) *and* $D^{[k+1]} = D^{[k]} = D = \textsc{Diag}(\mathbf{d})$ *with entries* $d_i = 1$ *if* $w_i \geq {}^{\lambda}\!/_{\rho}$,

$d_i = -1$ *if* $w_i \leq -\lambda/\rho$ *and* $d_i = 0$ *if* $-\lambda/\rho \leq w_i \leq \lambda/\rho$.

Proof. The proof of this lemma is similar to Lemma 3.2.1 and hence is omitted.

Spectral Properties and Regimes

We give the spectral properties of the ADMM matrix operator that we will use in our convergence analysis.

Lemma 4.2.2 $\|M^{[k]}\| = \|(I - (D^{[k]})^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2(D^{[k]})^2 - I)\| \leq 1$ *and the eigenvalues of* $M^{[k]}$ *lie in the closed disk in the complex plane with center* $1/2$ *and radius* $1/2$*, denoted as* $\mathcal{D}(1/2, 1/2)$*. In particular, if* $M^{[k]}$ *has any eigenvalue with absolute value* $\sigma(M^{[k]}) = 1$*, then that eigenvalue must be exactly 1. And eigenvalue* 1 *must have a complete set of eigenvectors (no Jordan blocks larger than* 1×1*).*

Proof. We temporarily omit the pass number $^{[k]}$. Observing $2D^2 - I$ is an orthogonal matrix by $(2D^2 - I)(2D^2 - I) = I$,

$$\begin{aligned} \|M\|_2 &= \|M(2D^2 - I)\|_2 \\ &= \|D^2 - I + (A^{\mathrm{T}}A/\rho + I)^{-1}\|_2 \\ &= \sigma(D^2 - I + (A^{\mathrm{T}}A/\rho + I)^{-1}) \leq 1 \end{aligned}$$

Besides,

$$\begin{aligned} \|M - 1/2 I\|_2 &= \|(1/2 I - D^2) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2D^2 - I)\|_2 \\ &= \| - 1/2 I(2D^2 - I) + \rho(A^{\mathrm{T}}A + \rho I)^{-1}(2D^2 - I)\|_2 \\ &= \|\rho(A^{\mathrm{T}}A + \rho I)^{-1} - 1/2 I\|_2 \|(2D^2 - I)\|_2 \\ &= \|(1/\rho A^{\mathrm{T}}A + I)^{-1} - 1/2 I\|_2 \end{aligned}$$

The eigenvalues of $(1/\rho A^{\mathrm{T}}A + I)^{-1}$ lie in the interval $(0, 1]$ and then the eigenvalues of

$(1/\rho A^{\mathrm{T}}A+I)^{-1}-1/2 I$ lie in the interval $(-1/2, 1/2]$, hence $\|M-1/2 I\|_2 \leq 1/2$. The eigenvalues of $M - 1/2 I$ lie in the closed circular disk on the complex plane with center 0 and radius $1/2$, denoted $\mathcal{D}(0, 1/2)$. The eigenvalues of M lie in the disk $\mathcal{D}(1/2, 1/2)$, which is entirely in the open right half plane plus the origin. In particular, if M has any eigenvalue with absolute value $\sigma(M) = 1$, then that eigenvalue must be exactly 1. Due to Theorem 3.2.2, eigenvalue 1 must have a complete set of eigenvectors. □

Similar to ISTA operator $\mathbf{R}_{aug}^{[k]}$ and FISTA operator $\mathbf{N}_{aug}^{[k]}$, the eigenvalues of the augmented matrix $\mathbf{M}_{aug}^{[k]}$ consist of those of $M^{[k]}$ plus an extra eigenvalue equal to 1. If $M^{[k]}$ already has an eigenvalue equal to 1, then the extra eigenvalue 1 may or may not add a corresponding eigenvector. This lets us to characterize regimes in terms of their possible Jordan canonical forms. Essentially the following lemma says that all the eigenvalues must have absolute value strictly less than 1, except for the eigenvalue equal to 1. And the eigenvalue 1's geometric multiplicity either equal to or one less than its algebraic multiplicity.

Lemma 4.2.3 *Assuming $D^{[k+1]} = D^{[k]}$, then $\mathbf{M}_{aug}^{[k]}$ in (4.2.8) is fixed and has a spectral decomposition $\mathbf{M}_{aug}^{[k]} = \mathbf{P}_M \mathbf{J}_M^{[k]} \mathbf{P}_M^{-1}$, where $\mathbf{J}_M^{[k]}$ is a block diagonal matrix*

$$\mathbf{J}_M^{[k]} = \textsc{Diag}\left\{ \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, I_M^{[k]}, \widehat{\mathbf{J}}_M^{[k]} \right\} \tag{4.2.9}$$

where any of these blocks might be missing. Here $I_M^{[k]}$ is an identity matrix and $\widehat{\mathbf{J}}_M^{[k]}$ is a matrix with spectral radius strictly less than 1.

Proof. The proof is similar to Lemma 3.3.2. Since $M^{[k]}$ satisfies Lemma 4.2.2 and hence contributes two blocks $I_M^{[k]}$ and $\widehat{\mathbf{J}}_M^{[k]}$. Embedding $M^{[k]}$ into the entire matrix yields a matrix $\mathbf{M}_{aug}^{[k]}$, with the exact same set of eigenvalues with the same algebraic and geometric multiplicities, except for eigenvalue 1.

If the upper left block of $\mathbf{M}_{aug}^{[k]}$ has no eigenvalue equal to 1, then $\mathbf{M}_{aug}^{[k]}$ has a simple eigenvalue 1. In general for eigenvalue 1, the algebraic multiplicity goes up by one and the geometric multiplicity can either stay the same or increase by 1. In other words, $\mathbf{M}_{aug}^{[k]}$ either satisfies the conditions of Lemma 3.3.1, or the algebraic and geometric multiplicities of eigenvalue 1 for $\mathbf{M}_{aug}^{[k]}$ differ by 1, meaning we have a single 2×2 Jordan block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$. □

Lemma 4.2.3 give rise to four "regimes" associated with the ADMM iteration, depending on the flag matrix and the eigenvalues of $\mathbf{M}_{aug}^{[k]}$. We treat separately the case where the flag matrix remains the same at each iteration and transitional case where the flag matrix changes.

When the flag matrix is unchanged from one step to the next: $D^{[k+1]} = D^{[k]}$, the ADMM operator $\mathbf{M}_{aug}^{[k]}$ remains invariant over those passes so that the convergence behavior can be categorized into three situations. In the following, we describe these three possible regimes distinguished by the eigenstructure of $\mathbf{M}_{aug}^{[k]}$ respectively.

[A]. The spectral radius of $M^{[k]}$ is strictly less than 1. The block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ is absent from (4.2.9), and the block $I_M^{[k]}$ is 1×1. In the case where the optimal solution exists and is unique, the result is linear convergence to a unique fixed point at a rate determined by the next largest eigenvalue in absolute value (largest eigenvalue of the block $\widehat{\mathbf{J}}_M^{[k]}$), according to the theory of the power method. If the flags $\widehat{D}^{[k]}$ are consistent with the eigenvector satisfying (4.2.5), then the eigenvector must satisfy the KKT conditions because of Lemma 4.2.1.

[B]. $M^{[k]}$ has an eigenvalue equal to 1 which results in a 2×2 Jordan block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ for $\mathbf{M}_{aug}^{[k]}$. Then the theory of power method implies that the vector iterates will converge to the invariant subspace corresponding to the largest eigenvalue 1. Therefore, the iteration

process tends to a constant step. If we satisfy the conditions for global convergence of ADMM, then such a flag change is guaranteed to occur.

[C]. $M^{[k]}$ has an eigenvalue equal to 1, but the block $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ is absent. If the flags $\widehat{D}^{[k]}$ are consistent with the eigenvector satisfying (4.2.5), the result is linear convergence. The convergence rate of this regime also depends on $\sigma(\widehat{\mathbf{J}}_M^{[k]})$. If unique solution is assumed, the iterations will jump out of this regime towards the end.

When the flag matrix does change, it means the set of active constraints at the current pass in the process has changed, and the current pass is a transition to a different operator with a different eigenstructure.

[D]. The operator $M^{[k+1]}$ will be different from $M^{[k]}$ due to different flag matrix.

4.2.2 Local Linear Convergence

We first invoke the global convergence property of ADMM.

Theorem 4.2.4 *Problem* (2.1.3) *has a solution if and only if there is a saddle point* $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ *of the Lagrangian* $\mathcal{L}_\rho = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{z}\|_1 + \mathbf{y}^{\mathrm{T}}(\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}\|_2^2$. *With* $\mathbf{u} = \mathbf{y}/\rho$, *let* $(\mathbf{x}^{[0]}, \mathbf{z}^{[0]}, \mathbf{u}^{[0]})$ *be any starting point and* $(\mathbf{x}^{[k]}, \mathbf{z}^{[k]}, \mathbf{u}^{[k]})$ *be the sequence generated by ADMM. Then for any* $k \geq 1$, $f(\mathbf{x}^{[k]}) + r(\mathbf{z}^{[k]}) \longrightarrow f(\mathbf{x}^*) + r(\mathbf{z}^*)$, *where* $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2$, $r(\mathbf{z}) = \lambda\|\mathbf{z}\|_1$. *In particular, the ADMM iterates for the LASSO problem should converge to the solution, i.e.* $\mathbf{x}^{[k]} \longrightarrow \mathbf{x}^*$, $\mathbf{z}^{[k]} \longrightarrow \mathbf{z}^*$, $\mathbf{u}^{[k]} \longrightarrow \mathbf{u}^*$.

Proof. The proof is omitted. We remark that the objective value convergence is a restatement of the convergence Theorem in [12]. The iterates convergence follows the Theorem 8 of [37]. □

Lemma 4.2.5 *Consider the matrix and eigenvector*

$$\mathbf{M}_{aug} = \begin{pmatrix} M & \mathbf{p} \\ 0 & 1 \end{pmatrix} \text{ and } \quad \mathbf{w}^*_{aug} = \begin{pmatrix} \mathbf{w}^* \\ 1 \end{pmatrix} \text{ such that } \quad \mathbf{M}_{aug}\mathbf{w}^*_{aug} = \mathbf{w}^*_{aug} \tag{4.2.10}$$

where M is any $n \times n$ matrix such that the spectral radius σ of M satisfies $\sigma(M) < 1$. Then the following holds.

[a] For any $\epsilon > 0$, there is a matrix norm $\|\cdot\|_P$ such that $\sigma(M) \leq \|M\|_P < \sigma(M) + \epsilon$. In particular, one can choose ϵ small enough so that $\|M\|_P < 1$. Also, there is a positive constant C_a (depending on M) such that for any vector of matrix X, $\|X\|_P \leq C_a\|X\|_\infty$ and $\|X\|_\infty \leq C_a\|X\|_P$.

[b] The iterates of the power iteration $\mathbf{w}^{[k+1]}_{aug} = \mathbf{M}_{aug}\mathbf{w}^{[k]}_{aug}$ satisfy $\|\mathbf{w}^{[k]}_{aug} - \mathbf{w}^*_{aug}\|_P \leq \|M\|^{[k]}_P \|\mathbf{w}^{[0]}_{aug} - \mathbf{w}^*_{aug}\|_P$ and hence converge linearly to $\mathbf{w}^*_{aug}$ at a rate bounded by $\sigma(M) + \epsilon$ where ϵ is the same arbitrary constant used in [a]. This is a special case of the theory behind the power method for computing matrix eigenvalues.

[c] Given any positive constant C_b, if $\mathbf{w}^{[0]}_{aug}$ is any vector such that $\|\mathbf{w}^{[0]}_{aug} - \mathbf{w}^*_{aug}\|_\infty \leq C_b/C_a^2$ then $\|\mathbf{M}^{[k]}_{aug}\mathbf{w}^{[0]}_{aug} - \mathbf{w}^*_{aug}\|_\infty \leq C_b$ for all k. In particular, if w^*_i is bounded away from two points $\pm\lambda/\rho$ and $C_b = \min_i\{|w^*_i - \lambda/\rho| - \epsilon, |w^*_i + \lambda/\rho| - \epsilon\} > 0$, then any element of vector $\mathbf{M}^{[k]}_{aug}\mathbf{w}^{[0]}_{aug}$ should be bounded away from two points $\pm\lambda/\rho$ for all $k = 0, 1, 2, \cdots$.

Proof. (a) and (b) are restatements of the Lemma 6.2 in [11]. For (c), we make more comments. Define $P_{aug} = \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}$ with the P from part (a), and define the corresponding P_{aug}-norm on the augmented quantities. Define the following balls around

the eigenvector $\mathbf{w}^*_{aug}$:

$$\begin{aligned}\mathcal{B}_a &= \{\mathbf{w}_{aug} : \|\mathbf{w}_{aug} - \mathbf{w}^*_{aug}\|_\infty \leq C_b\} \\ \mathcal{B}_b &= \{\mathbf{w}_{aug} : \|\mathbf{w}_{aug} - \mathbf{w}^*_{aug}\|_{Paug} \leq C_b/C_a\} \\ \mathcal{B}_c &= \{\mathbf{w}_{aug} : \|\mathbf{w}_{aug} - \mathbf{w}^*_{aug}\|_\infty \leq C_b/C_a^2\}\end{aligned} \tag{4.2.11}$$

From part(a), $\mathcal{B}_c \subseteq \mathcal{B}_b \subseteq \mathcal{B}_a$. From part(b), if any power method iterate satisfies $\mathbf{w}^{[0]} \in \mathcal{B}_b$, then all subsequent iterates stay in $\mathcal{B}_b$. Hence if the power method starts in $\mathcal{B}_c$, all subsequent iterates will lie in $\mathcal{B}_a$. □

Finally, we show one of our main results in this chapter.

Theorem 4.2.6 *Suppose the LASSO problem* (2.1.3) *has a unique solution and strict complementarity. Then eventually the ADMM iteration reaches a stage where it converges linearly to that unique solution.*

Proof. By Lemma 4.2.1, the strict complementarity is equivalent to $w_i^* \neq \pm\lambda/\rho$. Note that $w_i^{[k]}$ (where k is the pass number) could only be in one of three cases: $w_i^{[k]} > \lambda/\rho$, $w_i^{[k]} < -\lambda/\rho$ or $-\lambda/\rho < w_i^{[k]} < \lambda/\rho$. We can let $C_b = \min\{|w_i^* - \lambda/\rho| - \epsilon, |w_i^* + \lambda/\rho| - \epsilon\} > 0$ for a positive constant ϵ sufficiently small to make $C_b > 0$. By Theorem 4.2.4, there exists a pass K such that $\|\mathbf{w}^{[K]} - \mathbf{w}^*\|_\infty^2 < (C_b/C_a^2)$. Then By lemma 4.2.5(c), $\mathbf{w}^{[k]}$ for all $k > K$ lie in $\mathcal{B}_a$ stated in lemma 4.2.5(c). This means that $w_i^{[k]}$ will remain in one of three cases: $w_i^{[k]} > \lambda/\rho$, $w_i^{[k]} < -\lambda/\rho$ or $-\lambda/\rho < w_i^{[k]} < \lambda/\rho$ and will never change to another case for all $k > K$. Moreover, the flag matrix $D^{[k]} = \textsc{Diag}(\text{sign}(\text{Shr}_{\lambda/\rho}(\mathbf{w}^{[k]})))$ will also keep remained due to its definition. Therefore, starting at the K-th pass, the ADMM iteration reduces to the power method on the matrix $\mathbf{M}_{aug}^{[K]}$. It converge linearly at a rate of $\sigma(M^{[K]})$. □

4.3 Coordinate Descent for the LASSO Model

4.3.1 Local Linear Convergence

The cyclic coordinate descent essentially goes through and updates all of the components in a cyclic fashion instead of updating them simultaneously as the gradient descent method. To make difference from ADMM iterations, we denote $\mathbf{y}$ as the iterate of Coordinate Descent, $\mathbf{y}_i^{[k]}$, the i-th coordinate of $\mathbf{y}^{[k]}$ and A_i as the i-th column of A. All coordinates other than i are denoted as $-i$. Recall the Coordinate Descent updates for problem (2.1.3) in Section 4.1 is equivalently as follows.

Algorithm 15 One pass of Coordinate Descent for LASSO problem

Input: Start with $\mathbf{y}^{[k]}$.

1: **for** $i = 1, 2, \cdots, n$ **do**

2: $\quad \mathbf{y}_i^{[k+1]} = \mathrm{Shr}_\lambda \left(A_i \mathbf{b} - \sum_{j=1}^{i-1} (A^{\mathrm{T}} A)_{ij} \mathbf{y}_j^{[k+1]} - \sum_{j=i+1}^{|n|} (A^{\mathrm{T}} A)_{ij} \mathbf{y}_j^{[k]} \right) / \|A_i^{\mathrm{T}} A_i\|.$

3: **end for**

Output: $\mathbf{y}^{[k+1]}$ for next pass.

Lemma 4.3.1 *Suppose the LASSO problem has strict complementarity in Section 2.1.2. then there exists a K such that for all $k > K$, the CD iterate $\mathbf{y}^{[k]}$ is close enough to the optimal solution $\mathbf{y}^*$ that sign($\mathbf{y}^{[k]}$) is fixed.*

Proof. We first define the following index set based on $\mathbf{y}^*$

$$\mathcal{E} = \{i \in \{1, \cdots, n\} : \mathbf{y}_i^* \neq 0\} \tag{4.3.12}$$

and $\overline{\mathcal{E}}$ as the complement set of $\mathcal{E}$. Consequently,

$$\mathbf{y}^* = \begin{bmatrix} \mathbf{y}_{\mathcal{E}}^* \\ \mathbf{0} \end{bmatrix} \text{ (with } \mathbf{y}_{\mathcal{E}}^* \text{ all non-zero).}$$

Based on the notation, it also can be seen that $A_{\mathcal{E}}$ is composed of the columns corresponding to the nonzero element of $\mathbf{y}^*_{\mathcal{E}}$. For the simplicity of our analysis and without loss of generality, we split matrix A based on $\mathcal{E}$ and permute the columns so that $A = [A_{\mathcal{E}}, A_{\overline{\mathcal{E}}}]$. Then the optimality condition of problem (2.1.3) then can be rewritten as

$$\begin{aligned} &\text{(a)} \quad A^T_{\mathcal{E}}\mathbf{b} - A^T_{\mathcal{E}}A_{\mathcal{E}}\mathbf{y}^*_{\mathcal{E}} - A^T_{\mathcal{E}}A_{\overline{\mathcal{E}}}\mathbf{y}^*_{\overline{\mathcal{E}}} = \lambda \mathbf{d}_{\mathcal{E}} \\ &\text{(b)} \quad A^T_{\overline{\mathcal{E}}}\mathbf{b} - A^T_{\overline{\mathcal{E}}}A_{\mathcal{E}}\mathbf{y}^*_{\mathcal{E}} - A^T_{\overline{\mathcal{E}}}A_{\overline{\mathcal{E}}}\mathbf{y}^*_{\overline{\mathcal{E}}} = \lambda \mathbf{d}_{\overline{\mathcal{E}}}, \end{aligned}$$

where $\mathbf{d}_{\mathcal{E}}$ is composed of elements $\{\pm 1\}$ based on $\text{sign}(\mathbf{y}^*_{\mathcal{E}})$, and $\mathbf{d}_{\overline{\mathcal{E}}}$ is composed of elements strictly between -1 and $+1$ by the assumption of strict complementarity. Let δ be the largest entry in absolute value found in $\mathbf{d}_{\overline{\mathcal{E}}}$ so that

$$\text{Shr}_\lambda \left(A^T_{\overline{\mathcal{E}}}\mathbf{b} - A^T_{\overline{\mathcal{E}}}A_{\mathcal{E}}\mathbf{y}^*_{\mathcal{E}} - A^T_{\overline{\mathcal{E}}}A_{\overline{\mathcal{E}}}\mathbf{y}^*_{\overline{\mathcal{E}}} \right) = 0.$$

Now consider the case of $\overline{\mathcal{E}}$. Let $\mathbf{y}^{[k]}$ be a vector very close to $\mathbf{y}^*$ such that $\|\mathbf{y}^{[k]} - \mathbf{y}^*\|_\infty < \epsilon_1$. Then $|\mathbf{y}_l - \mathbf{y}^*_l| < \epsilon_1 \ \forall l \in \mathcal{E}$ and $|\mathbf{y}_j| < \epsilon_1 \ \forall j \in \overline{\mathcal{E}}$. If ϵ_1 is sufficiently small, then (b) above induces

$$\|A^T_{\overline{\mathcal{E}}}A_{\mathcal{E}}\mathbf{y}^{[k]}_{\mathcal{E}} - A^T_{\overline{\mathcal{E}}}A_{\overline{\mathcal{E}}}\mathbf{y}^{[k]}_{\overline{\mathcal{E}}} - A^T_{\overline{\mathcal{E}}}\mathbf{b}\|_\infty < \lambda(\delta + c_1\epsilon_1) < \lambda,$$

so that $\text{Shr}_\lambda(\mathbf{y}^{[k]}_{\overline{\mathcal{E}}}) = 0$. Here c_1 is some constant magnification factor depending only on A. Since $\mathbf{y}^{[k]}$ converges to $\mathbf{y}^*$, it would imply that the future iterates, starting from $\mathbf{y}^{[k]}$, would have zeros in the $\overline{\mathcal{E}}$ part. Next consider the case of $\mathcal{E}$. Let $\epsilon_2 = \min_{\mathcal{E}}\{|\mathbf{y}^*_{\mathcal{E}}|\} - c_2$ for a positive constant c_2 sufficiently small to make $\epsilon_2 > 0$. And for each $l \in \mathcal{E}$, we define a ball around each $\mathbf{y}^*_l$ that $\mathcal{B}(\mathbf{y}^*_l) = \{\mathbf{y}_l : \|\mathbf{y}_l - \mathbf{y}^*_l\|_\infty \leq \epsilon_2\}$.

According to Theorem 16 in [107], cyclic coordinate descent converges to LASSO problem at the rate of $O(1/k)$. Hence there must exist an iteration number K that for

all $k > K$,

$$\|\mathbf{y}^{[k]} - \mathbf{y}^*\|_\infty \leq \epsilon = \min\{\epsilon_1, \epsilon_2\}$$

which implies $\mathbf{y}_{\overline{\mathcal{E}}}^{[k]} = 0$ and $\mathbf{y}_l^{[k]} \in \mathcal{B}(\mathbf{y}_l^*)$, $\forall l \in \mathcal{E}$. In other words, for each component i, $\mathbf{y}_i^{[k]}$ must fall in one of three cases: $\mathbf{y}_i < 0$, $\mathbf{y}_i = 0$ and $\mathbf{y}_i > 0$ and never jump out to another case. □

Theorem 4.3.2 *Suppose the optimal solution* $\mathbf{y}^*$ *of problem* (2.1.3) *is sparse such that* $A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}}$ *is positive definite, with* $\mathcal{E}$ *defined in* (4.3.12)*. Then eventually the cyclic coordinate descent iteration reaches a stage where it converges linearly to the solution.*

Proof. From Lemma 4.3.1, when close enough to the optimal solution, $\mathrm{sign}(\mathbf{y}^{[k]})$ is fixed. So $\mathbf{y}_{\overline{\mathcal{E}}}^{[k]}$ remained zero for future iterations. As for $\mathbf{y}_{\mathcal{E}}^{[k]}$, we can simplify the resulting Coordinate Descent updates in Alg. 3 by eliminating $\mathbf{y}_j^{[k]}$ $(\forall j \in \overline{\mathcal{E}})$ so that $\forall l \in \mathcal{E}$,

$$\begin{aligned} \mathbf{y}_l^{[k+1]} = \Big(A_l \mathbf{b} - \lambda \widehat{\mathbf{d}}_l^{[k]} - \textstyle\sum_{j=1}^{l-1} (A^{\mathrm{T}} A)_{lj} \mathbf{y}_j^{[k+1]} \\ - \textstyle\sum_{j=l+1}^{|\mathcal{E}|} (A^{\mathrm{T}} A)_{lj} \mathbf{y}_j^{[k]} \Big) / \|A_l^{\mathrm{T}} A_l\|. \end{aligned} \tag{4.3.13}$$

Assuming λ is large enough such that optimal solution $\mathbf{y}^*$ is sparse and $A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}}$ is positive definite, then (4.3.13) is equivalent to the Gauss-Seidel method applied to

$$A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}} \mathbf{y}_{\mathcal{E}} = (A^{\mathrm{T}} \mathbf{b})_{\mathcal{E}} - \lambda \mathbf{d}_{\mathcal{E}} \tag{4.3.14}$$

where elements of $\mathbf{d}_{\mathcal{E}}$ is fixed and equal to either 1 or -1. The iteration must converge linearly by the theory of Gauss-Seidel method [51]. □

Remark: we note here that $A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}}$ being positive definite is a mild assumption in practice. Since the optimal solution is sparse, $|\mathcal{E}| < m$ can be satisfied easily for λ not too small, and in many applications this is sufficient to guarantee that all columns of $A_{\mathcal{E}}$

are linearly independent. If not, one can increase λ to increase the sparsity.

4.3.2 Comparison between Coordinate Descent and ISTA

In this part, we show that Coordinate Descent should converge faster than ISTA when both Coordinate Descent and ISTA iterations are in their final regimes from the viewpoint of preconditioning. The next lemma shows the equivalence of the ISTA iteration and the classical Richardson iteration [72].

Lemma 4.3.3 *When ISTA iteration reaches the final regime, the regime of linear convergence, then the ISTA iteration is equivalent to the Richardson iteration for solving the linear system* (4.3.14).

Proof. When ISTA reaches the final regime, according to Theorem 3.4.3 $\widehat{\mathbf{w}} = (I - {}^{1}\!/_{L} A^{\mathrm{T}}A)\widehat{\mathbf{x}} + {}^{1}\!/_{L} A^{\mathrm{T}}\mathbf{b}$ would fall in into three cases: $\widehat{\mathbf{w}} < -{}^{\lambda}\!/_{L}$, $-{}^{\lambda}\!/_{L} < \widehat{\mathbf{w}} < {}^{\lambda}\!/_{L}$, $\widehat{\mathbf{w}} > {}^{\lambda}\!/_{L}$, and never jump out to another case. Hence the shrinkage operator in updating step is fixed so that ISTA reduces to

$$\widehat{\mathbf{x}}_{\mathcal{E}}^{[k+1]} = (I - {}^{1}\!/_{L} A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}})\widehat{\mathbf{x}}_{\mathcal{E}}^{[k]} + {}^{1}\!/_{L}((A^{\mathrm{T}}\mathbf{b})_{\mathcal{E}} - \lambda \widehat{\mathbf{d}}_{\mathcal{E}}) \tag{4.3.15}$$

where $\widehat{\mathbf{d}}_{\mathcal{E}} = \mathrm{sign}(\mathbf{x}^{*}_{\mathcal{E}})$. And $\widehat{\mathbf{x}}_{\overline{\mathcal{E}}}^{[k+1]} = 0$. The resulting ISTA updates (4.3.15) is exactly Richardson iteration [72] for solving (4.3.14). □

Combining the result of Theorem 4.3.2 and Lemma 4.3.3, if both Coordinate Descent and ISTA reach their final regimes, then Coordinate Descent is equivalent to Gauss-Seidel iteration and ISTA is equivalent to Richardson iteration for solving the same linear system (4.3.14). Let T and U be the diagonal and strict upper triangular part of $A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}}$ and L is the lipschitz constant. Gauss-Seidel iteration can be written as the preconditioned

Richardson iteration as below (and hence generally faster):

$$\mathbf{y}^{[k+1]} = (I - (T + U^{\mathrm{T}})^{-1} A_{\mathcal{E}}^{\mathrm{T}} A_{\mathcal{E}}) \mathbf{y}^{[k]} + (T + U^{\mathrm{T}})^{-1} ((A^{\mathrm{T}}\mathbf{b})_{\mathcal{E}} - \lambda \mathbf{d}_{\mathcal{E}})$$

with preconditioner $L(T + U^{\mathrm{T}})^{-1}$ [72].

4.4 Numerical Examples

We use the compressive sensing examples to illustrate ADMM and Coordinate Descent convergence behavior. Moreover, we also compare them with ISTA and FISTA in the previous chapter.

The basic setting is similar to Example 2 in Section 3.6. Here is a brief review. Suppose there exists a true sparse signal represented by a n-dimensional vector $\mathbf{x}_s$ with s non-zero elements. We observe the image of $\mathbf{x}_s$ under the linear transformation $A\mathbf{x}_s$, where A is the so-called measurement matrix. Our observation thus should be

$$\mathbf{b} = A\mathbf{x}_s + \epsilon \tag{4.4.16}$$

where ϵ is the observation noise. The goal is to recover the sparse vector $\mathbf{x}_s$ from the measurement matrix A and observation $\mathbf{b}$. We use the LASSO model (2.1.3) and let $A \in \mathbb{R}^{n \times p}$ be Gaussian matrix whose elements are *i.i.d* distributed as $\mathcal{N}(0, 1)$, ϵ be a vector whose elements are *i.i.d* distributed as $\mathcal{N}(0, \sigma^2)$ with $\sigma = 10^{-3}$.

4.4.1 Examples of ADMM and Coordinate Descent

Figure 4.1 (First) shows the convergence behavior of ADMM on the same Example 2 in Section 3.6, in which $A \in \mathbb{R}^{128 \times 1024}$, $s = 10$ and $\lambda = 1$. In the ADMM iteration, we let the parameter in the augmented Lagrangian ρ to be 100. It costs ADMM 151

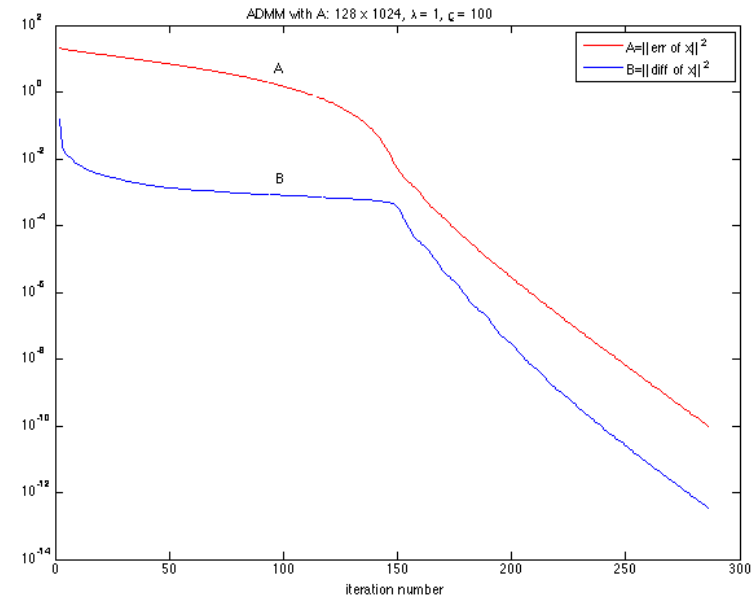
iterations to reach the final regime and eventually converges in totally 286 iterations. It can be seen that curves of the difference of iterates remain at a constant number. This is because they are in the constant regime [B] such that the difference between consecutive iterates (curves B) are converging to a constant vector. Finally, it shows the linear convergence. The rate is the second largest eigenvalue of $\mathbf{M}_{aug}$, which equals to 0.946. Figure 4.1 (Second) shows the eigenvalues of the operators $\mathbf{M}_{aug}$ during the final regime. The eigenvalues for $\mathbf{M}_{aug}$ lie close to the boundary but strictly inside the circle $\mathcal{D}(1/2, 1/2)$ (except for 0 and 1), consistent with Lemma 4.2.2.

Figure 4.2 (First) shows the convergence behavior of CD on the same Example 2 in Section 3.6, in which $A \in \mathbb{R}^{128\times 1024}$ and $s = 10$. It costs CD 114 iterations to reach the final regime and converges in totally 119 iterations. Though there are only 5 iterations in the final regime, the linear convergence is clear in the figure, with the rate 0.184. Figure 4.2 (Second) shows the spectrum of CD.

4.4.2 Comparison of All Scalable Methods

In this section, we use the compressive sensing problems to compare all mentioned scalable methods, including ISTA, FISTA, ADMM and Coordinate Descent. The numerical results are summarized in Table 4.1 and Table 4.2. We compare the convergence behavior in terms of the total number of iterations (Total #), number of iterations to reach final linear regime (Final #) and the linear rate in the final regime (Rate), with different λ and sparsity s. We also report the behavior of a hybrid F/ISTA that follows FISTA during initial iterations to reach the final regime and then switch to ISTA until convergence. In general, after comparison with the linear rates, we see that, Coordinate Descent is often much faster than the other iterations during the final linear regime.

Especially, the examples in Sections 3.6 & 4.4.1 come from the instance marked ** in Table 4.2. To compare all of the scalable algorithms together, we illustrate their



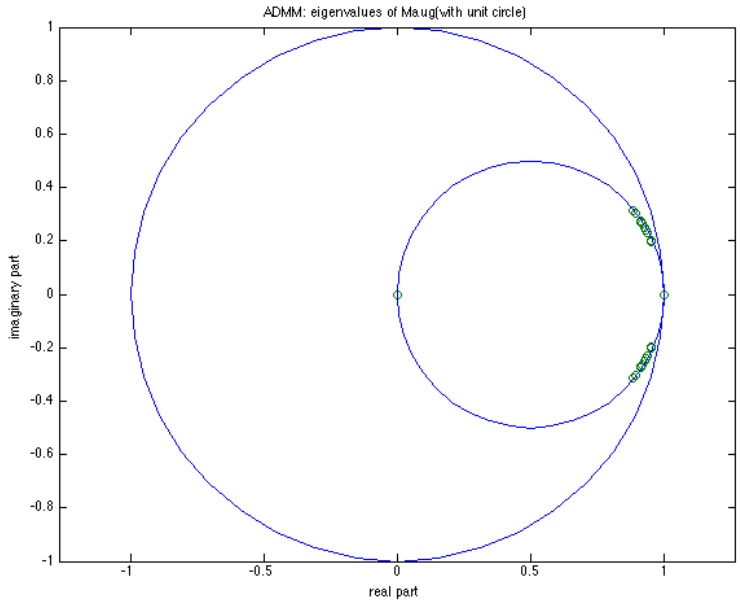


Figure 4.1: (First): ADMM on Example 2 of Section 3.6. Curves **A**: $\|\mathbf{x}^{[k]} - \mathbf{x}^*\|$. **B**: $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|$. (Second) Spectrum of ADMM operator $\mathbf{M}_{aug}$ on the complex plane during the last regime of the iteration process. The unit circle and $\mathcal{D}(1/2, 1/2)$ are shown for reference.

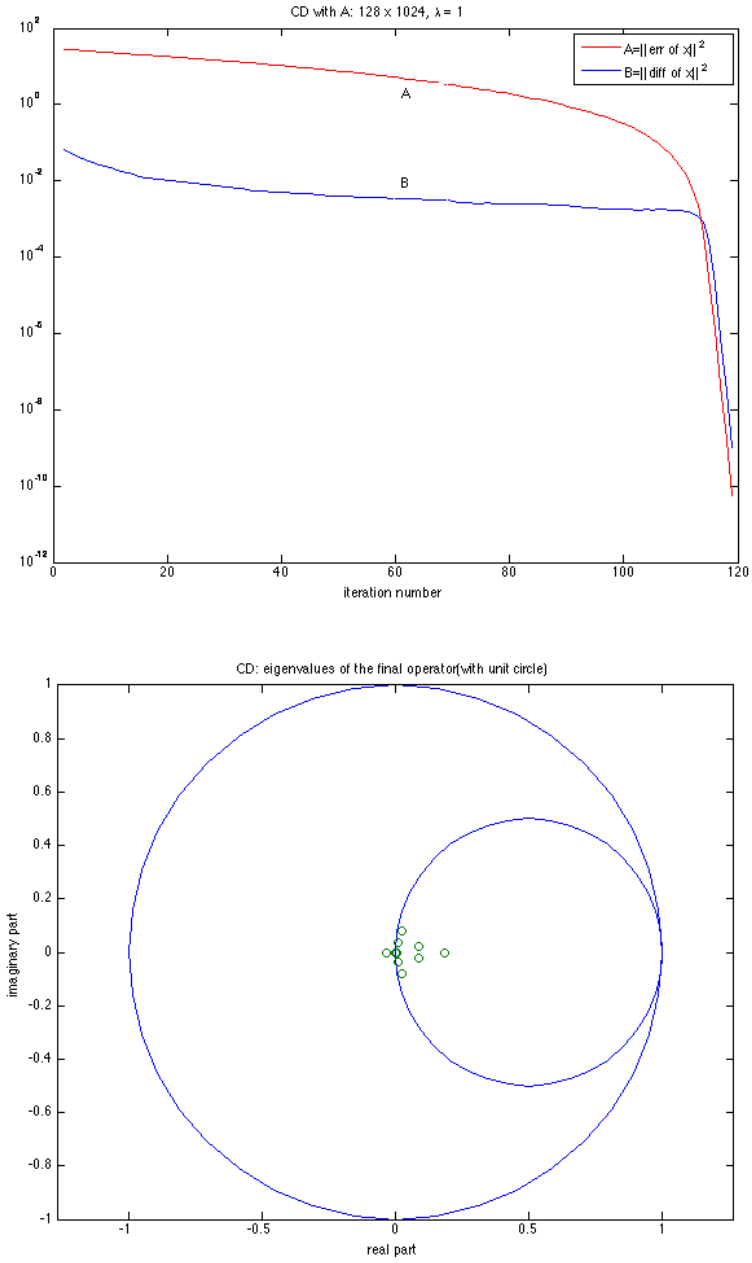


Figure 4.2: (First): CD on Example 2 of Section 3.6. Curves **A**: $\|\mathbf{x}^{[k]} - \mathbf{x}^*\|$. **B**: $\|\mathbf{x}^{[k]} - \mathbf{x}^{[k-1]}\|$. (Second): Spectrum of CD operator during linear regime. The unit circle and $\mathcal{D}(1/2, 1/2)$ are shown for reference.

performances on one figure. (See Figure 4.3.) Figure 4.3 (First) shows the error of iterates of all algorithms. Observe that all of them pass through a few transitions in the early part of the iterations and then settle on their final linear regimes. Figure 4.3 (Second) shows the eigenvalues during the final regimes for all algorithms.

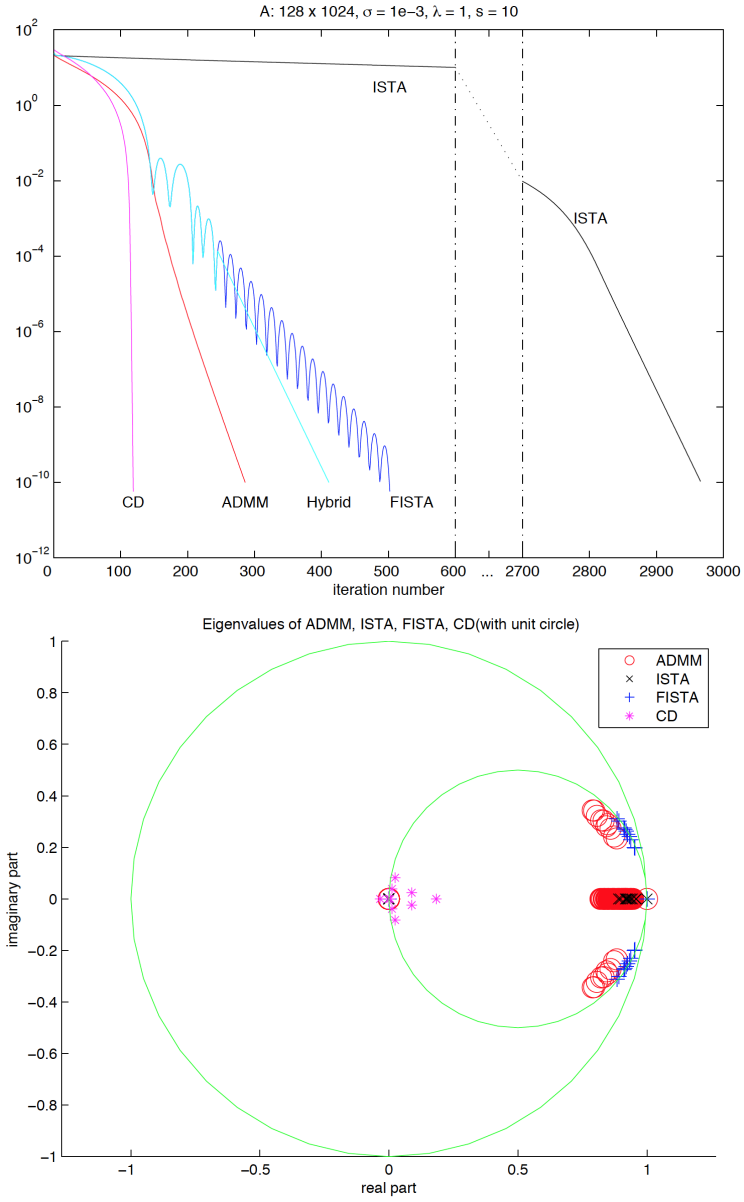


Figure 4.3: (First): Convergence behavior in terms of error of iterates of ADMM, ISTA, FISTA, Hybrid F/ISTA and CD for the instance marked ** in Table 4.2. (Second): Spectrum of operators of all iterations on the left during the final regime. The unit circle and $\mathcal{D}(1/2, 1/2)$ on the complex plane are shown for reference.

Problem Setting	ADMM	ISTA	FISTA	F/ISTA	CD
Dimension 64×512	$s = 7, \lambda = 0.3$				
Total #	758	6824	541	512	399
Final #	711	6749	402	402	394
Convergence Rate	0.897	0.958	0.975	0.958	0.141
Dimension 64×512	$s = 7, \lambda = 1$				
Total #	271	2143	393	313	124
Final #	216	2062	226	226	119
Convergence Rate	0.897	0.958	0.973	0.958	0.141
Dimension 64×512	$s = 7, \lambda = 5$				
Total #	116	583	289	216	28
Final #	44	438	84	84	25
Convergence Rate	0.899	0.954	0.960	0.954	0.076
Problem Setting	ADMM	ISTA	FISTA	F/ISTA	CD
Dimension 64×512	$s = 14, \lambda = 0.3$				
Total #	943	10000	1226	2193	1011
Final #	904	-	961	961	952
Convergence Rate	0.963	-	0.996	0.996	0.878
Dimension 64×512	$s = 14, \lambda = 1$				
Total #	447	9321	1367	2211	350
Final #	268	7713	546	546	294
Convergence Rate	0.963	0.996	0.995	0.996	0.878
Dimension 64×512	$s = 14, \lambda = 5$				
Total #	286	2927	1054	1532	102
Final #	87	1561	758	758	63
Convergence Rate	0.962	0.994	0.995	0.994	0.822

Table 4.1: Examples of compressive sensing with dimension 64×512. λ is the parameter in problem (2.1.3) and s is the number of non-zero elements of optimal solution. (Total #): the total number of iterations with maximum 10^4. (Final #): number of iterations before reaching final linear regime. (Rate): The linear rate (eigenvalue) in the final regime.

Problem Setting	ADMM	ISTA	FISTA	F/ISTA	CD
Dimension 128×1024	$s = 10, \lambda = 0.3$				
Total #	611	9390	693	622	383
Final #	498	9273	507	507	377
Convergence Rate	0.946	0.959	0.976	0.959	0.184
Dimension 128×1024	$\boxed{**}\, s = 10, \lambda = 1 \,\boxed{**}$				
Total #	286	2966	502	411	119
Final #	151	2823	245	245	114
Convergence Rate	0.946	0.959	0.973	0.184	0.878
Dimension 128×1024	$s = 10, \lambda = 5$				
Total #	193	776	298	266	32
Final #	33	596	99	99	24
Convergence Rate	0.946	0.959	0.959	0.184	0.822

Problem Setting	ADMM	ISTA	FISTA	F/ISTA	CD
Dimension 128×1024	$s = 25, \lambda = 0.3$				
Total #	1760	10000	2024	2306	2821
Final #	1541	-	2021	2021	2754
Convergence Rate	0.980	-	0.997	0.996	0.881
Dimension 128×1024	$s = 25, \lambda = 1$				
Total #	807	10000	1777	2245	893
Final #	566	-	1724	1724	820
Convergence Rate	0.980	-	0.997	0.996	0.883
Dimension 128×1024	$s = 25, \lambda = 5$				
Total #	544	7315	1528	2017	237
Final #	225	5857	863	863	19
Convergence Rate	0.980	0.996	0.996	0.996	0.892

Table 4.2: Examples of compressive sensing with dimension 128×1024. λ is the parameter in problem (2.1.3) and s is the number of non-zero elements of optimal solution. (Total #): the total number of iterations with maximum 10^4. (Final #): number of iterations before reaching final linear regime. (Rate): The linear rate (eigenvalue) in the final regime. $\boxed{**}$: The instance illustrated in Fig. 4.3.

Chapter 5

Structured Sparse Inverse Covariance Estimation

In previous chapters, we look into the LASSO model and study convergence of different scalable methods on the sparsity structure. In real world, many applications encounter the situation where the data pattern has richer structure. In this chapter, we will explore the group sparse structure introduced in Section 1.2.2. We propose a novel estimator that can exploit such structure for estimating inverse covariance matrix discussed in Section 2.4.

The organization of this chapter is as follows. Section 5.1 reviews the standard sparse inverse covariance problem, existing popular methods, and their issues. In Section 5.2, we define a novel group norm for matrix, given prior knowledge of structural information. The group is described as either a principal submatrix or the matrix diagonal. Our estimator is proposed in Section 5.3. The solution matrix is then the sum of these possibly overlapping submatrices. Section 5.4 describes an efficient way of applying the Frank-Wolfe method for this problem. The resulting algorithm can decompose into separate eigenvalue computations on the smaller submatrices at each iteration, with improvement

for both performance and computation cost. In Section 5.5 we present simulation results, showing that the benefit of exploiting group structure significantly improves sample complexity. Finally, we show the performance of our model on congressman voting data sets in Section 5.6 and the stock datasets in Section 5.7.

In next chapter, we extend and show that our estimator can cover a wide class of more general group structured models.

5.1 Introduction and Motivation

The sparse inverse covariance estimation arises in network reconstruction, finance, machine learning. It gains significant popularity because the sparsity pattern of inverse covariance gives the conditional independence between each pair of variables. However, in most of the applications, the number of samples is much fewer than dimensions. In particular, assume $\hat{C} \in \mathbb{S}^p$ is the sample covariance generated by n samples, where $n \ll p$. then the sample covariance matrix $\hat{C}$ is not invertible, and the pseudoinverse $\hat{C}^\dagger$ is inaccurately dense, as we have discussed in detail in Section 2.4.

The goal of the so-called sparse inverse covariance estimation is matrix is try to compute the sparse inverse of $\hat{C}$ under $n \ll p$. The most popular method to do is known as graphical LASSO (G-LASSO) estimator [7, 129].

$$\begin{array}{ll} \underset{X}{\text{minimize}} & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) + \rho\|X\|_1 \\ \text{s.t.} & X \succeq 0 \end{array} \tag{5.1.1}$$

for some regularization parameter $\rho > 0$. Recall our general form (1.1.1), $f(X) = -\text{logdet}(X) + \text{tr}\,(\hat{C}X)$, $r(X) = \rho\|X\|_1$ and $\mathcal{D} = \{X : X \succeq 0\}$. In fact $f(X)$ is the maximum likelihood function. $r(X)$ is used to induce sparsity. By adding $r(X)$ and $\mathcal{D}$, the effective degrees of freedom are reduced, and it has been shown that the

resulting estimator has a much lower sample complexity than inverting $\hat{C}$. However, this estimator does not incorporate any prior structural knowledge from the problem domain. Additionally, solving (5.1.1) is computationally challenging in general if p is large. We describe these two issues below in detail.

Computational Issue

Most existing methods for solving (5.1.1) require a sequence of eigenvalue decompositions (EDs) [8, 29, 45, 105, 131]. This is expensive if p is large; a dense ED requires $O(p^3)$ computations, and sparse EDs (like Lanczos type methods) are not suitable when the full eigenvalue spectrum is needed. (Used this way, they may be far slower than dense methods.) There are some exceptions; for example [109] updates each row in a block coordinate descent fashion, and maintains inverses using only rank-2 updates; [28] uses chordal decomposition to compute Newton steps efficiently in an interior point solver; and [85] uses neighborhood selection, which enforces the conditional independence condition one variable at a time. These methods are more or less intuitive, relying on general convex optimization principles; however, their scalability is limited. On the other end of the spectrum is BIG-QUIC [66] which can solve up to 1 million variables. This breakthrough method simultaneously makes estimates of the matrix sparsity while also optimizing for it, and updating via block coordinate descent with carefully chosen (non-principal) submatrices. However, it demonstrates the tradeoff between simplicity and scalability; there are many intricate details for a successful implementation.

Prior Structural Knowledge

In many applications, prior knowledge of data pattern is roughly known in advance. A key question arising is how to use them effectively. In the recent decade, there has been growing interest in the statistics community to exploit group structure in the estimators [23, 88, 90]. For example, [84] proposes thresholding the sample covariance matrix in order to identify fully-connected components of the graphical model, effectively decomposing

(5.1.1). More recently, [63] learns overlapping submatrix groups probablistically and penalizes accordingly. To our knowledge, this is the only work that addresses overlapping group sparsity in matrices; however, repeated full EDs are still needed to find the inverse covariance estimate.

All above motivate us to find an estimator that can both computationally efficient and exploit prior structure knowledge for sparse inverse covariance estimation.

5.2 Matrix Group Norm

For an index set $\gamma \subset \{1, \ldots, p\}$ and a vector $x \in \mathbb{R}^p$, define x_γ as the subvector of x indexed by γ; for the reverse, define the augmenting linear map $\gamma : \mathbb{R}^{|\gamma|} \to \mathbb{R}^p$ such that

$$(A_\gamma u)_\gamma = u, \quad (A_\gamma u)_i = 0 \text{ if } i \notin \gamma.$$

In [90], the overlapping group norm is defined as the solution to

$$\|x\|_G \;=\; \min_{u_1,\ldots,u_l} \left\{ \sum_{k=1}^{l} w_k \|u_k\|_2 \;:\; x = \sum_{k=1}^{l} A_{\gamma_k} u_k \right\} \tag{5.2.2}$$

for nonnegative weights $w_1, \ldots, w_l$. (A common choice is $w_k = |\gamma_k|^{-1}$.) Used as a penalty term or in a constraint, this norm is shown to promote group structure; a small subset of index sets γ_k are "active", and $x_i = 0$ whenever i is not in an active set.

We extend this concept to matrices, by defining groups implicitly through index sets $\beta \subset \{1, \ldots, p\}$, where $X_{\beta,\beta}$ is the submatrix of X selected by the rows and columns indicated by β. Let $\mathbb{S}^p$ denote the set of $p \times p$ matrices. We define $\mathcal{A}_\beta : \mathbb{S}^{|\beta|} \to \mathbb{S}^p$ such that

$$(\mathcal{A}_\beta(U))_{\beta,\beta} = U, \quad \mathcal{A}_\beta(U)_{i,j} = 0 \text{ if } i \notin \beta \text{ or } j \notin \beta$$

and extend the overlapping group norm as the solution

$$\|X\|_G := \begin{cases} \underset{v,U_1,\ldots,U_l}{\text{minimize}} & w_0\|v\|_2 + \sum_{k=1}^{l} w_k\|U_k\|_F \\ \text{subj. to} & X = \mathbf{diag}(v) + \sum_{k=1}^{l} \mathcal{A}_{\beta_k}(U_k). \end{cases} \tag{5.2.3}$$

Here, X can be considered as the sum of smaller principal submatrices U_k and a diagonal term v, and w_k are nonnegative scalar weights. Note that the affine constraint imposes a sparsity pattern on X; if X does not adhere to this pattern (*e.g.* $X_{ij} \neq 0$ for $i, j \notin \beta_k, \forall k$) then we define $\|X\|_G = \infty$.

5.3 Inverse Covariance Estimation with Group Structure

Our goal is to develop a novel estimator that can make use of the prior structural knowledge and meanwhile speed up the computing. To accomplish it, we first propose following two models

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) \\ \text{s.t.} \quad & \|X\|_G \leq \alpha, \quad X \succeq 0 \end{aligned} \tag{5.3.4}$$

$$\begin{aligned} \text{minimize} \quad & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) + \tau\|X\|_G \\ \text{s.t.} \quad & X \succeq 0 \end{aligned} \tag{5.3.5}$$

where $\|X\|_G$ is defined in (5.2.3) and α and τ are parameters for the tuning parameters. The objective function remains the same as graphical LASSO since this is the maximum likelihood function given sample covariance matrix. The regularization $\|X\|_G$ can be either put in the constraint or in the objective function. The above two formulation can deal with group structure as desired. However, the computation with $X \succeq 0$ is still there.

So instead, we propose to solve the following model with regularization in the constraint.

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) \\ \text{s.t.} \quad & X = \sum_{k=1}^{l} \mathcal{A}_{\beta_k}(U_k) + \mathbf{diag}(v) \\ & w_0\|v\|_2 + \sum_{k=1}^{l} w_k\|U_k\|_F \leq \alpha \\ & U_k \succeq 0,\ k = 1, \ldots, l, \quad v_i \geq 0,\ i = 1, \ldots, p. \end{aligned} \tag{5.3.6}$$

where α is the penalized parameter to be tuned. Throughout this chapter, we denote estimator (5.3.6) as Norm-regularized Graphical LASSO (NG-LASSO). We make several important remarks as follows.

Implicit Sparsity

As defined, the constraint $\|X\|_G \leq \alpha$ restricts X to be within a sparsity pattern defined by the groups β_k. Specifically, we say that X is constrained to *sparsity pattern* if

$$X_{ij} = 0,\ \forall (i,j) \notin E. \quad \text{for} \quad E = \{(i,j) \mid i \neq j, i \in \{1, \ldots, p\},\ j \in \{1, \ldots, p\}\}. \tag{5.3.7}$$

The sparsity pattern induced by the group norm $\|X\|_G$ is

$$E(\{\beta_k\}_1^l) = \{(i,j) \mid i \neq j, \exists \beta_k \supset \{i,j\}\}.$$

PSD Constraint and Chordal Sparsity

Apart from group norm, another main difference between standard graphical LASSO (5.1.1) and our model (5.3.6) is the conic constraint (in particular the positive semidefinite (PSD) constraint). We admit a PSD constraint on each small group, i.e. $U_k \succeq 0, k = 1, \ldots, l$ instead of on the whole matrix $X \succeq 0$. As defined in (5.3.7), a sparsity pattern E can be equivalently interpreted as the edge set of an undirected graph with p nodes.

We say E is *chordal* if any cycle of length over three has a chord (an edge joining two non-consecutive vertices in the cycle). The following key property relates chordal graphs to the sparse PSD constraint.

Theorem 5.3.1 *[1, 56] ([57] for the dual version) If a matrix X has chordal sparsity E, corresponding to an undirected graph with maximal cliques (complete subgraphs) $\beta_1, \ldots, \beta_l$, then*

$$X \succeq 0 \iff X = \sum_{k=1}^{l} \mathcal{A}_k(U_k), \quad U_k \succeq 0,\ k = 1, \ldots l.$$

By this theorem, if the sets $\beta_1, \ldots, \beta_l$ are the maximal cliques of a chordal graph, then the last constraint in (5.3.6) are equivalent to $X \succeq 0$ restricted to sparsity pattern $E(\{\beta_k\}_1^l)$. This suggests our estimator can be equivalent to

$$\begin{aligned} \text{minimize} \quad & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) \\ \text{s.t.} \quad & \|X\|_G \leq \alpha, \quad X \succeq 0 \end{aligned} \tag{5.3.8}$$

when chordal sparsity condition is satisfied.

Handling Nonchordal Sparsity

For a nonchordal sparsity pattern E, a *chordal extension* is any chordal sparsity pattern $\bar{E} \supset E$. Several efficient methods exist for finding $\bar{E}$ for general E [122]. this chordal completion $\bar{E}$ will correspond to a new set of maximal cliques (the new groups). In general, the number of added edges $|\bar{E} \setminus E|$ may be significant, and may degrade with the intuitive group structure derived from domain knowledge. However, for special cases, there may exist an intuitive chordal completion that does not significantly alter the group assignments.

5.4 Frank-Wolfe Method and Complexity Analysis

We apply Frank-Wolfe method for problem (5.3.6). Frank-Wolfe method has regained much attention in minimizing sparse problems [68], mimicking greedy approaches yet having guaranteed optimality for convex problems. The Frank-Wolfe algorithm for solving $\min_X\{f(X) : X \in \mathcal{D}\}$ is described in Alg. 16. Using step sizes $\eta^{[t]} = 2/(t+2)$, it is

Algorithm 16 One Pass of Frank-Wolfe algorithm

Input: $X^{[t]} \in \mathcal{D}$, step size $\eta^{[t]}$, at t-th iteration.

1: Compute gradient $\nabla f(X^{[t]})$.
2: Compute forward step : $S = \arg\min_{S\in\mathcal{D}}\langle S, \nabla f(X^{[t]})\rangle$.
3: Update primal variable : $X^{[t+1]} = (1-\eta^{[t]})X^{[t]} + \eta^{[t]}S$.

Output: optimal $X^{[t+1]}$.

known that the iterates of algorithm 16 converge at the sublinear rate of $O(1/t)$ [35, 44]. We refer readers to Section 1.3 for details of this method.

Forward Step

At each iteration, the forward step consists of l parallelizable projections on positive definite matrices. Specifically, at each forward step, we compute

$$U_j^* = \frac{\alpha}{w_j\|Z_j\|_F}Z_j, \quad U_k = 0,\ \forall k \neq j.$$

where index $j = \arg\max_k w_k^{-1}\|Z_k\|_F$ and $Z_j = \mathbf{proj}_{\mathcal{C}_j}(-\nabla f(X)_{\beta_j,\beta_j})$. Then $S = \sum_k w_k \mathcal{A}_{\beta_k,\beta_k}(U_k)$.

Gradient Computation

Another operation that needs to take into account is computing logdet(X) and X^{-1}. In general, to compute the gradient $\nabla(\log\det(X)) = X^{-1}$ requires matrix inversion, which negates the computational complexity gain by decomposing the PSD cone. However, if the groups β_k form a chordal pattern, we use the fast projected inverse method of [2, 82]

which require at each step l inversions of matrices at most of order $|\beta_k|$. In summary, this method takes a matrix X with chordal sparsity E and its associated clique tree, and computes $\mathbf{proj}_E(X^{-1})$ by taking inverses of matrices of order

$$|\eta_k|, \quad k = 1, \dots l, \quad \eta_k := \beta_k \setminus \beta_{\mathrm{par}(k)}$$

where $\mathrm{par}(k)$ is the parent of clique k in the given clique tree. By adding $\mathrm{logdet}(D_k)$ of intermediary matrices $D \in \mathbb{S}_+^{|\eta_k|}$, the $\mathrm{logdet}(X)$ can be simultaneously efficiently computed. Both steps have complexity $O(|\beta_k|^3)$ per group.

Algorithm 17 One step of Frank-Wolfe algorithm for (5.3.6)

Input: $X^{[t]} \in \mathcal{D}$, t-th iteration, step size $\eta := \frac{2}{t+2}$

1: Find $\nabla f(X) = X^{-1} + C$.

2: Find the forward direction U^+:

$$Z_0 = \mathbf{proj}_{\mathbb{R}^p_+}(-\mathbf{diag}(\nabla f(X))).$$
$$Z_k = \mathbf{proj}_{\mathbb{S}_+^{|\beta_k|}}(-\nabla f(X)_{\beta_k,\beta_k}).$$
$$j = \arg\max_k w_k^{-1} \|Z_k\|_F.$$
$$U_j^+ = \frac{\alpha}{w_j \|Z_j\|_F} Z_j, \quad U_k^+ = 0, \ \forall k \neq j.$$

3: Update $X^{[t+1]} = X^{[t]} + \frac{2}{t+2} U^+$

Output: $X^{[t+1]}$.

Applying both techniques, Alg. 17 describes the procedure for one iteration to find the NG-LASSO estimator (5.3.6). All above discussion leads to the following results.

Proposition 5.4.1 *If $|\beta_k| \leq p/l$, then the per-iteration complexity of the proposed algorithm grows as $O(p^3/l^2)$.*

Proof. The main computational bottleneck at each step is a sequence of EDs of the submatrices $\nabla f(X)_{\beta_k,\beta_k}$, both for inverting X and for projecting on the PSD cone. For both operations, the complexity is $O(|\beta_k|^3)$ per group. If $|\beta_k| < p/l$ excluding the diagonal group, then the total per-iteration complexity of the proposed optimization

procedure has a per-iteration complexity of $O(p^3/l^2 + p)$, and much smaller than $O(p^3)$ for G-LASSO. □

5.5 Numerical Simulations

Here we show two simulation results: one on a general group sparsity pattern, and the other on a specific chordal pattern (banded). To pick α and ρ, we sweep powers of two in $\{2^{-5}, \ldots, 2^5\}$ and picked the best performing ρ or α for each test. In cases where the best performing ρ or α is on the boundary, additional parameters were tested until this is no longer the case.

5.5.1 Baselines

As one baseline, we solve (5.1.1). However, since group structure also reveals matrix sparsity, for fair comparison we also solve (5.1.1) restricted to the sparsity pattern induced by the groups:

$$\begin{array}{ll} \underset{X}{\text{min.}} & -\text{logdet}(X) + \text{tr}\,(\hat{C}X) + \rho\|X\|_1 \\ \text{s.t.} & X \succeq 0 \\ & X \in \mathbf{B} := \{X \mid X_{ij} = 0 \text{ if } i, j \notin \beta_k \; \forall k\}, \end{array} \tag{5.5.9}$$

which we call restricted group LASSO (RG-LASSO). We solve these baselines using the Douglas-Rachford method [27, 80] for minimizing the sum of m convex functions. Note that this ADMM-type of method has the advantage over the projected gradient method because it has no step size restrictions based on the objective function's Lipschitz constant, which for log det function is unbounded. To be specific,

$$f_1(X) = -\text{logdet}(X) + \text{tr}\,(\hat{C}X), \quad f_2(X) = \rho\|X\|_1$$

and f_3, f_4 as indicator functions for constraints

$$f_3(X) = \begin{cases} 0 & X \succeq 0 \\ \infty & \text{else} \end{cases}, \quad f_4(X) = \begin{cases} 0 & X \in \mathbf{B} \\ \infty & \text{else} \end{cases}.$$

The proximal operator [86] for a convex function $f(X)$ is defined as

$$prox_f(Z) = \arg\min_X f(X) + (1/2)\|X - Z\|_F^2$$

and is defined for all Z, even if Z is not in the domain of f. (This is especially useful for $f = \log\det$ and $Z \not\succeq 0$.) From optimality conditions, it can be shown that

$$prox_{tf_1}(Z) := V\mathbf{diag}(q)V^T, \quad 2q_i = -d_i + \sqrt{d_i^2 + 4t}$$

where $V\mathbf{diag}(d)V^T$ is the eigenvalue decomposition of $t\hat{C} - Z$. Similarly, $prox_{tf_2}$ is the shrinkage operator, and $prox_{tf_3}$, $prox_{tf_4}$ are projections on their respective constraint spaces.

5.5.2 Random Sparsity

For $X \in \mathbb{S}^p$, we randomly select l groups $\beta_k \subset \{1, \dots p\}$ of size b, and assume that this is the known group structure. Additionally, we select $\lceil \sigma_G \cdot l \rceil$ "active" groups (for $0 < \sigma_G < 1$); the identity of these groups are not known in training. In this simulation, we investigate the sample size required to recover the active groups, comparing G-LASSO, RG-LASSO, and NG-LASSO. In general, the sparsity patterns here are not chordal. To determine the sparsity pattern of X, *i.e.* the estimate of C^{-1}, we threshold so that

$$i, j \text{ is a nonzero index if } \frac{|X_{ij}|}{\max_{kl} |X_{kl}|} > \theta,$$

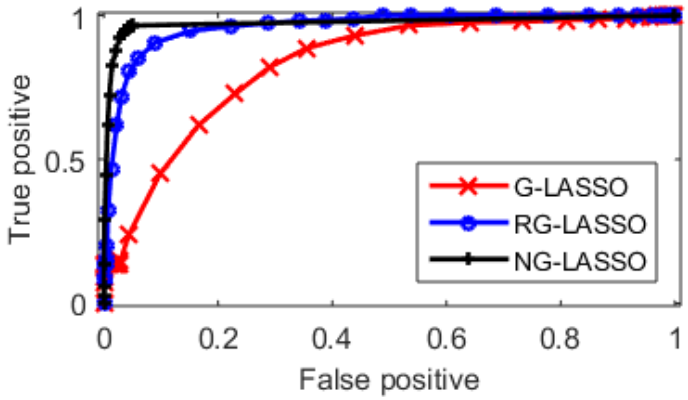


Figure 5.1: ROC curve for random block patterns where $p = 100$, $n = 25$. For each estimator, α, ρ picked to maximize AUC (area under this curve).

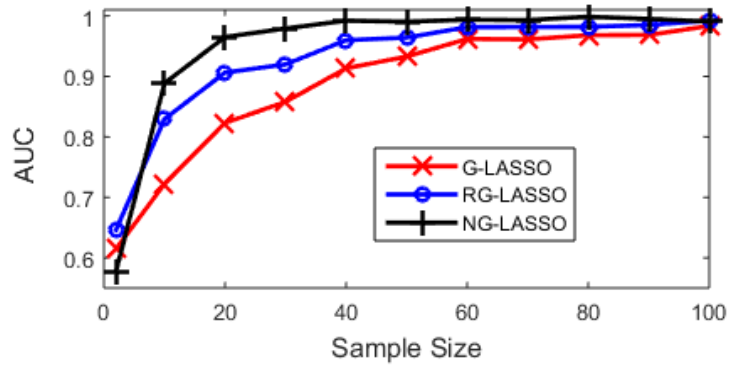


Figure 5.2: AUC for growing sample sizes (n) for random block patterns, averaged over 20 trials. $p = 100$, $l = 100$, $b = 5$ and $\sigma_G = 0.1$.

for $\theta \in [0, 1]$. Figure 5.1 shows the receiver operating characteristic (ROC) curve for $p = 100$ variables and $n = 25$ samples, where θ is implicitly swept. It is clear that NG-LASSO outperforms both other estimators; however, it is also evident that restricting the sparsity already offers much improvement. In order to remain agnostic to the right choice of θ, we use the area under the ROC curve (AUC) as the primary metric of estimator quality. Figure 5.2 plots the AUC for varying sample sizes n. Again, it is clear that for most cases, NG-LASSO outperforms RG-LASSO and G-LASSO. The exception is at very small values of n, where the number of observations is too low.

5.5.3 Banded Sparsity

Next, we present a more extensive experiment with a chordal sparsity pattern; specifically, a banded pattern. For $X \in \mathbb{S}^p$, we assume that the true sparsity pattern consists of a nonzero diagonal and some active diagonal blocks of size b, where b is known but the true sparsity pattern is not. This gives in total $l = p - b + 1$ candidate groups $\beta_k = \{k, \cdots, k+b\}$ for $k = 1, \cdots, l$. Among l groups, we assume $\sigma_G l$ groups are active (where $0 < \sigma_G < 1$). Moreover, we simulate in-group sparsity; that is, for $0 < \sigma_I \leq 1$, we fix $\Pr(X_{ij} \neq 0 | i, j \in \beta_k) = \sigma_I$. Note that using only known information, we must assume the sparsity pattern is banded with bandwidth b.

We construct an invertible C^{-1} with the true sparsity pattern, and form a sample covariance matrix $\hat{C}$ by sampling from a multivariate Gaussian with zero mean and covariance C. The goal is to use the estimators to correctly recover the sparsity pattern of C using $\hat{C}$ where the number of observations n is as small as possible.

Figure 5.3 shows a small example when $p = 100, n = 50$ and $\sigma_I = 0.25$. There are in total 90 groups with bandwidth 10 in the banded sparsity pattern, where the 9 active groups (true sparsity) are in blue. We pick the estimator nonzeros by thresholding on the absolute value, choosing the threshold to, in each case, maximize min{# true positives, # true negatives}.

It is clear that, for this small example, G-LASSO yields many spurious nonzeros. By simply restricting the sparsity pattern to $\mathbf{B}$, the performance of RG-LASSO already improves significantly, but NG-LASSO is still the best, since it accounts for sparsity in group selection as well.

Table 5.1 gives the result of a more extensive experiment for several p, n and σ_I, where the threshold, α, and ρ are picked to maximize the AUC. One can see that NG-LASSO is comparable with RG-LASSO when $p \approx n$, but is consistently better for $p \gg n$; both, however, are considerably improved over G-LASSO.

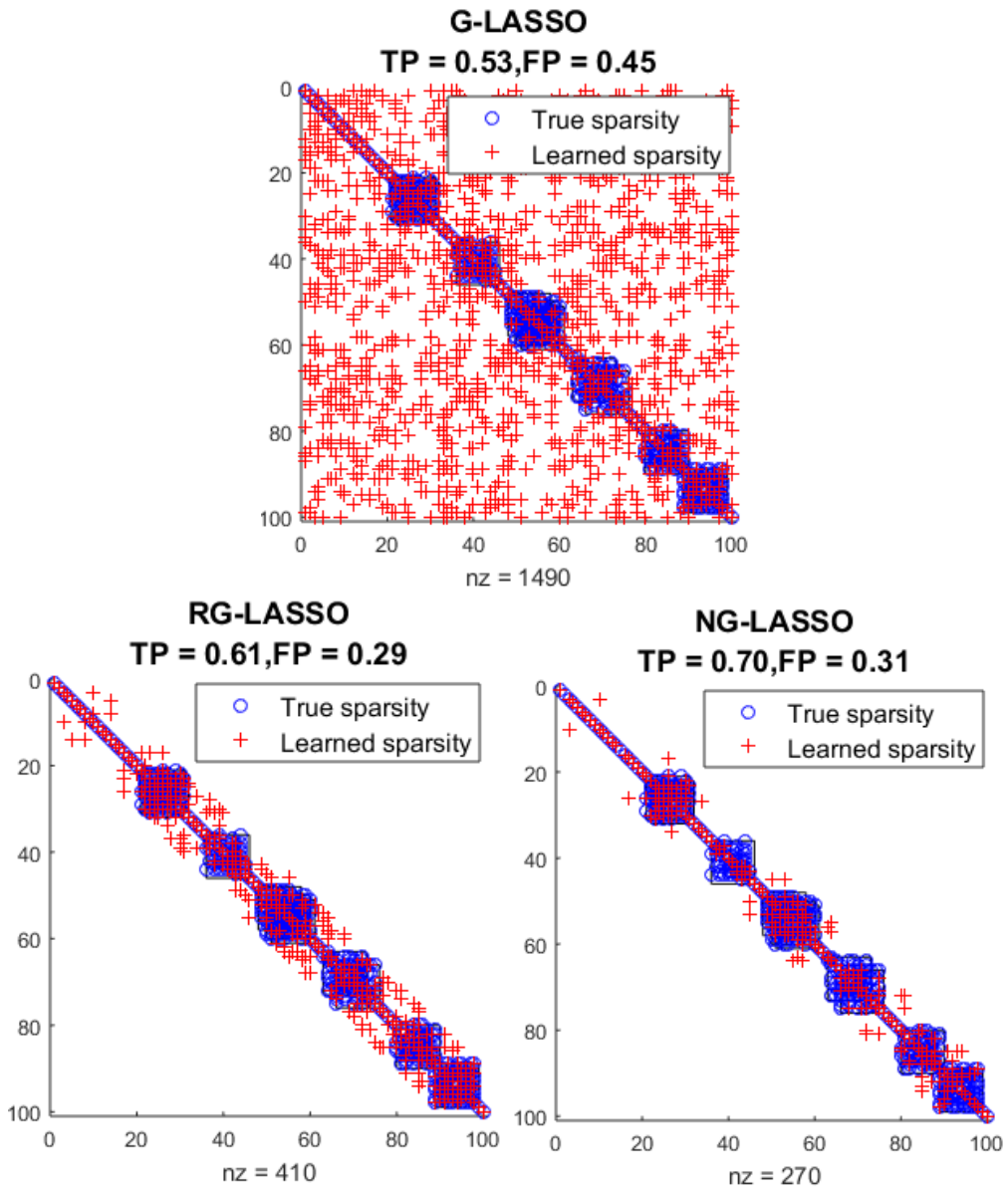


Figure 5.3: Banded pattern sparse inverse covariance estimation for $p = 100, n = 50$. From left to right are G-LASSO (5.1.1), RG-LASSO (5.5.9) and NG-LASSO (5.3.8). TP = true positive, FP = false positive.

p	n	σ_I	$\hat{C}$	$\hat{C}^\dagger$	G	RG	NG
100	10	0.1	0.43	0.44	0.50	0.57	**0.65**
100	10	0.25	0.39	0.40	0.48	0.58	**0.64**
100	100	0.1	0.59	0.60	**0.89**	0.88	0.88
100	100	0.25	0.52	0.69	0.83	0.80	**0.85**
1000	10	0.1	0.41	0.41	0.49	0.52	**0.63**
1000	10	0.25	0.34	0.34	0.40	0.38	**0.68**
1000	100	0.1	0.55	0.56	0.56	0.58	**0.89**
1000	100	0.25	0.48	0.49	0.48	0.52	**0.90**

Table 5.1: Best AUC scores for $p \times p$ matrices with n samples and block sparsity σ_I. G = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. Bolded are the best estimators. $\hat{C}, \hat{C}^\dagger$ = AUC score using sampled $\hat{C}, \hat{C}^\dagger$ directly.

Table 5.2 gives the per-iteration and total runtime of the three methods. In all cases, the per-iteration runtime depends only on p and b, and for larger p, NG-LASSO enjoys a much smaller per-iteration runtime. Of course, the total runtime (influenced also by the number of iterations) is also important; however, our experiments suggest this value is much less predictable. Several trends do emerge, though; for all methods, the runtime grows significantly when n is very small or σ_I is very low, suggesting that problem conditioning influences convergence rate as well. However, the key takeaway is that for very large p it is impossible to maintain full EDs for each iteration, and some decomposition must be used.

5.6 Applications on Congressman Voting History Data

We take congress member voting record as real datasets for an application. The goal is to investigate the extreme views between various congress members. We scrape data from `https://www.govtrack.us/developers/data`. There are totally $12,611$ congress members make $23,750,842$ votes on $104,657$ votes. For each vote, there are a number of possible responses: `yea`, `nay`, `present`, `absent`, `abstain`, to name a few. We represent

		Per Iteration			Overall		
p	n	G	RG	NG	G	RG	NG
100	10	<0.1	<0.1	<0.1	**0.94**	3.93	7.3
100	100	<0.1	<0.1	<0.1	0.19	**0.17**	0.39
1000	10	1.6	1.6	**0.35**	**93.6**	108.69	351.1
1000	100	1.4	1.4	**0.35**	**13.8**	9.97	349.5
2500	100	21.3	19.6	**1.1**	852.9	556.7	**333.9**
5000	100	150.9	125.8	**1.7**	-	-	-

Table 5.2: Runtimes (in seconds) for $p \times p$ matrices with n samples, bandwidth $p/10$, and block sparsity $\sigma_I = 0.1$. G = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. For $p \leq 100$, ρ and α are the same as those used in Table 5.1. For $p = 2500$, we just run $\rho = 0.125$ and $\alpha = 32$, which was observed to work well for smaller p. For $p = 5000$, we only give runtimes for 1 iteration, to illustrate the growing gap in per-iteration runtime.

each vote as a 1-sparse vector of length equal to the number of voting choices for that vote. We then concatenate each vote vector to form one long voter vector x_i. The sample covariance matrix is then computed as

$$\hat{C}_{ij} = (x_i - \bar{x})^T (x_j - \bar{x}), \quad \bar{x} = \frac{1}{p} \sum_{i=1}^{p} x_i.$$

A large value of $\hat{C}_{ij}$ indicates that voter i and voter j vote similarly on the exact same motions.

We examine the performance of G-LASSO and NG-LASSO to see the effect of incorporating group information. Group structure for congress members can be constructed in two ways. One is to take members of the same gender(age, party, house/senate, state) as a group. The other way is to do clustering over the historic dataset and take congress members of the same clusters as groups. Since there is no way to have ground truth in real applications, we implement different group structures and evaluations.

Experiment A

We consider 101th - 114th US congress (1991-2015) voting on the specific hot topic

	Sample covariance		G-LASSO		NG-LASSO	
	nonzero	zero	nonzero	zero	nonzero	zero
C_{GT} nonzero	$5.6 \cdot 10^4$	$7.0 \cdot 10^4$	$1.1 \cdot 10^5$	$1.2 \cdot 10^4$	$9.1 \cdot 10^4$	$3.5 \cdot 10^4$
C_{GT} zero	$7.2 \cdot 10^3$	$1.0 \cdot 10^5$	$1.0 \cdot 10^5$	$8.3 \cdot 10^3$	$2.5 \cdot 10^4$	$8.4 \cdot 10^4$
	Precision	Recall	Precision	Recall	Precision	Recall
	0.89	0.44	0.53	0.90	0.79	0.72

Table 5.3: the confusion matrix of original sample covariance, G-LASSO and our estimator NG-LASSO.

"abortion". The groups are gender and party. As a comparison, we use the ratings given by NARAL Pro-Choice America (r_1) and National Right to Life Committee (NRLC) (r_2). The sparsity of $C_{GT} = \frac{1}{2}(r_1 r_1^T + r_2 r_2^T)$ captures grouping of people with extreme views. After filtering voters to only those with both NARAL and NRLC ratings, we have in total 483 voters (p) and 184 votes (n).

Through Table 5.3, we illustrate the performance of original sample covariance, G-LASSO and our estimator NG-LASSO. Our estimator obtains better precision (with a drop in recall) over G-LASSO. Additionally, Figure 5.4 shows the overlay in sparsity patterns using the threshold that maximizes the F1 score for each test. Our estimator obviously selects more correct sparsity.

Experiment B

We look into the 105th - 109th US congress (1997-2007). By cleaning and reordering the data, Figure 5.5 shows the potential group structure of congress members. Blue parts are reordered sample data. Red parts are structured pattern based on Democratic/Republican Party, Senate/House Representative, State. Each congress member may belong to multiple groups, so that groups could overlap with each other.

As for comparison, we take 105-108th congress as training data and compare our learned result with the 109th congress data. We also show the performance of G-LASSO for reference. On the left of Figure 5.6, we take party, representative type and state as

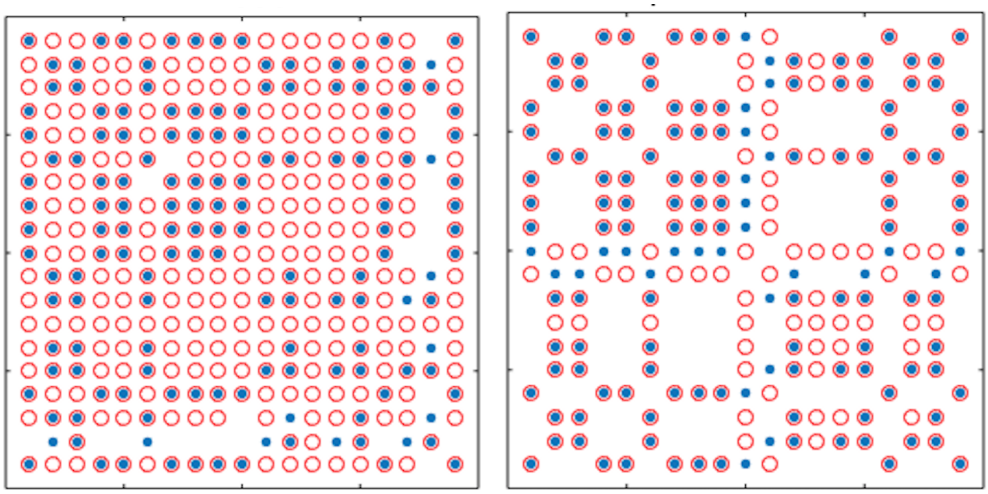


Figure 5.4: Sparsity pattern (rows/columns uniformly downsampled by 25×) of C_{GT} overlapped with nonzeros of the learned sparse inverse covariance matrix. Blue part is C_{GT}. Red part is G-LASSO(Left) and NG-LASSO(Right).

groups. On the right, we use K-means over historic dataset to obtain clusters and take them as groups. It can be seen that our estimator NG-LASSO can outperform G-LASSO when the groups are well picked.

5.7 Applications on Yahoo! Finance Data

In this section, we examine the performance of G-LASSO, RG-LASSO, and NG-LASSO in estimating the inverse covariance matrix for 14910 stocks, over a period of 100 or fewer days. This problem arises in finance known as *Markowitz portfolio optimization* [83], which discusses optimal portfolio allocations using only mean and covariance information. Specifically, when the objective is to minimize return volatility, it is advised to invest in each stock a weight w_i of one's assets, where w minimizes the following quadratic

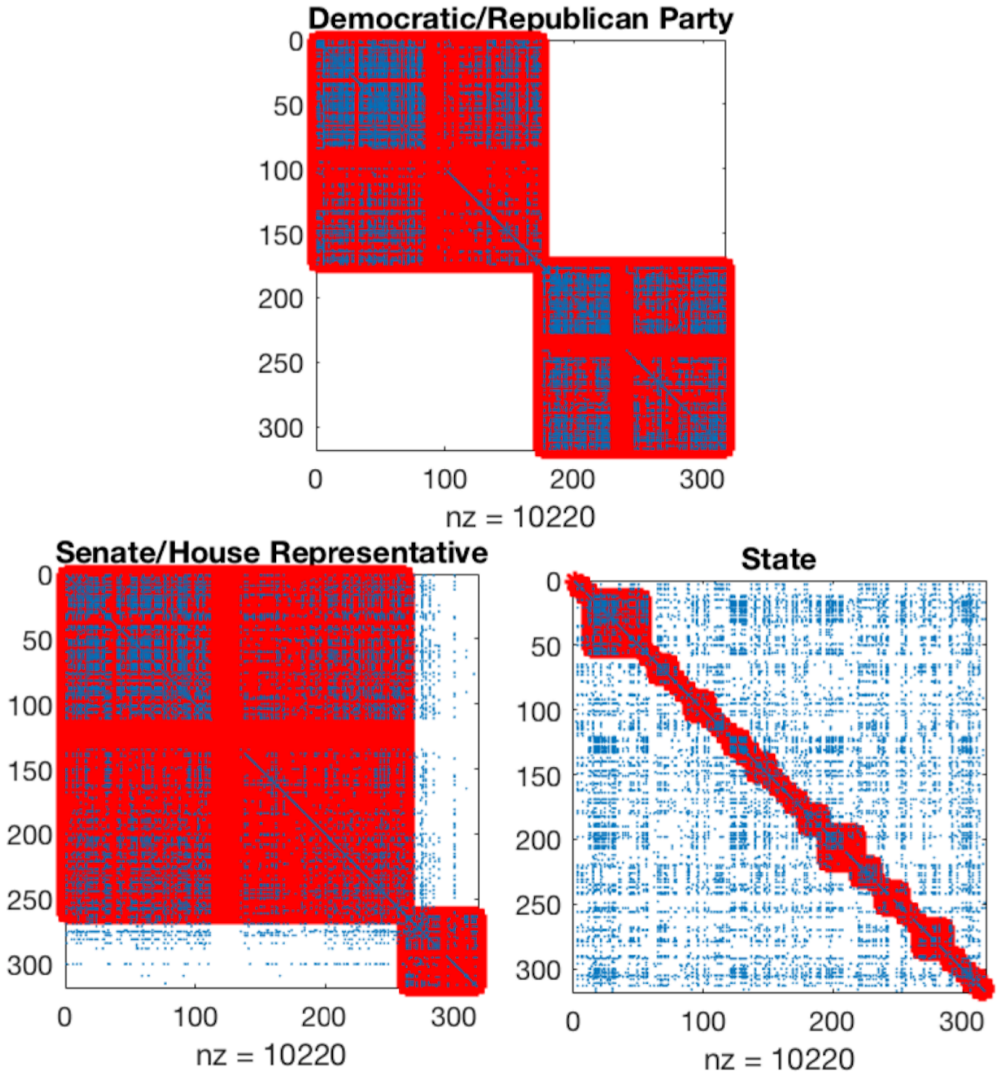


Figure 5.5: Voting samples from 105-109th(1997-2007) congress. Blue parts are reordered sample data. Red parts are structured pattern based on Democratic/Republican Party, Senate/House Representative, State.

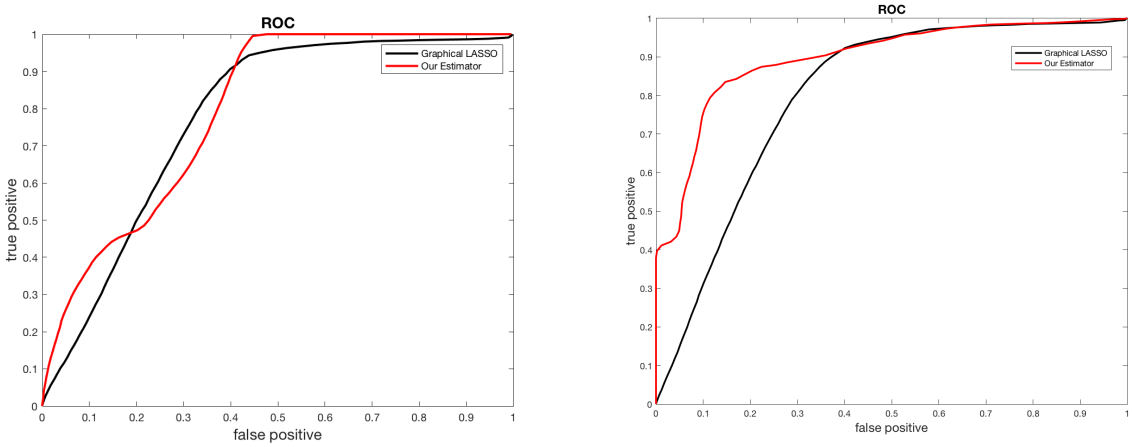


Figure 5.6: ROC of G-LASSO in black and NG-LASSO in red. (Left) Groups are party, representative type and state. (Right) Groups are clusters over historic data.

optimization problem

$$\min_{w} \quad w^T C w \qquad \text{s.t.} \quad \textstyle\sum_i w_i = 1$$

with optimal solution $w = (\mathbf{1}^T C^{-1} \mathbf{1})^{-1} C^{-1} \mathbf{1}$. However, in practice it is very difficult to apply this principle to large markets, especially when the amount of available historical data is limited, as the exact problem that C^{-1} is hard to estimate when p (thousands of stocks) is much greater than n (at most a few hundred days' opening or closing prices); using Markowitz' method in this regime can consistently underperform much simpler methods [31]. Thus, there is great interest in applying sparse estimators for a more generalizable calculation.

We attempt to estimate the inverse covariance matrix for stocks obtained from Yahoo! Finance, using daily closing prices as features. Define u_i as the full length observation vector for stock i, and S_i as the set of indices of u_i where that stock price observation is available. Note that not every day's value is provided for every stock. Define $V = \{1, 2, \ldots, 100\}$, $T = \{101, 102, \ldots, 200\}$, and $R = \{201, 202, \ldots, 200 + n\}$

as the indices of a validation, test, and train set respectively, and for all stocks i, $V_i = V \cap S_i, T_i = T \cap S_i, R_i = R \cap S_i$. The sample covariance is then calculated as

$$\hat{C}_{ij} = \frac{1}{|R_i \cap R_j|} \sum_{k \in R_i \cap R_j} u_i[k] u_j[k].$$

We solve (5.1.1), (5.5.9), and (5.3.6) sweeping ρ, α for powers of 2 from 2^{-5} to 2^5, using cross validation to pick the best ρ and γ. If the best ρ or α is on the boundary, additional powers of 2 are computed until this is no longer the case.

Unlike the controlled simulation, we have no real understanding of the ground truth, so we test the estimator's generalizability, by measuring the maximum likelihood over the test samples. Specifically, we compute the negative log likelihood (NLL) for precision matrix X and samples $\{u_i\}_{i \in T}$ in the test set:

$$\text{NLL} = -\log\det(X) + \sum_{i,j} \frac{X_{ij}}{|T_i \cap T_j|} \sum_{k \in T_i \cap T_j} u_i[k] u_j[k].$$

In this case, the smaller the value, the better.

Table 5.4 gives the test NLL for various p and n. As p grows (and especially as p/n grows) it is clear that G-LASSO has poorer performance, though the behavior is not as consistent as in Table 5.1. Additionally, it seems that restricting the sparsity pattern gives the regularization needed to achieve very good performance, though we note that for very large p, NG-LASSO still does better.

Table 5.5 gives the runtime of each experiment for the best set of parameters, showing 1) the time to do one set of EDs ($p \times p$ for LASSO and sum of $|\beta_k| \times |\beta_k|, k = 1, \ldots l$ for group methods), 2) the average per-iteration runtime ($\approx$ 1 ED) and 3) the total runtime. Since in this application all groups are nonoverlapping (all stocks are assigned a single sector and industry) computing RG-LASSO can be done in a fully decomposable

			sectors		industry	
p	n	G	RG	NG	RG	NG
100	10	**2.2e2**	2.5e2	7.0e2	2.7e2	5.5e2
500	10	2.0e3	1.8e3	3.7e3	**1.3e3**	3.5e3
500	100	1.3e3	**9.1e2**	4.1e3	9.3e2	3.1e3
1000	10	**2.5e3**	2.6e3	7.9e3	2.8e3	6.1e3
1000	100	1.1e13	**4.4e3**	7.1e3	**4.4e3**	7.4e3
2500	100	-	**1.1e4**	3.9e4	**1.1e4**	1.7e4
5000	100	-	1.5e12	1.1e7	5.9e10	**7.9e4**
14910	100	-	-	8.6e12	-	**5.2e5**

Table 5.4: Best test negative log likelihood for different methods, varying the number of stocks (p) and observations (n). G = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. '-' = runtime was too long.

manner, and can run at the same per-iteration speed as NG-LASSO. However, because the Douglas-Rachford method requires keeping 9 copies of the variables as separate iterates, it runs out of memory at $p = 14910$. Therefore our method still maintains the scalability advantage.

p	500	1000	2500	5000	14910
$p \times p$ ED	< 0.1	3.4e-1	5.2	4.3e1	1.1e3
S-ED	< 0.1	< 0.1	1.4e-1	6.4e-1	1.5e1
I-ED	< 0.1	< 0.1	< 0.1	< 0.1	1.0
G 1 it.	0.381	1.33	20.5	150.5	-
RG (S) 1 it.	< 0.1	0.20	1.6	6.0	-
RG (I) 1 it.	< 0.1	**0.17**	0.74	2.3	-
NG (S) 1 it.	< 0.1	0.20	1.4	5.5	79.6
NG (I) 1 it.	< 0.1	0.18	**0.72**	**2.2**	**20.2**
G all	20.0	1324.2	-	-	-
RG (S) all	3.0	88.9	713.6	**34.7**	-
RG (I) all	**2.5**	74.0	341.5	2351.1	-
NG (S) all	**2.5**	29.8	169.1	560.7	80377.2
NG (I) all	4.6	**26.9**	**79.7**	677.2	**2149.5**

Table 5.5: Runtimes (in seconds) of various algorithms for different matrix sizes ($\hat{C}$ is $p \times p$). S = sectors. I = industries. S-ED (I-ED) = time to compute $p_i \times p_i$ ED where $p_1, \ldots, p_l$ are the sizes of the l sector (industry) groups. G = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. '-' = runtime was too long.

Chapter 6

General Structured Model

In the previous chapter, we develop a group norm regularized estimator on the model problem known as sparse inverse covariance estimation. The matrix version of group structure can both exploit the prior knowledge and reduce computation cost. The content of this chapter would be the future directions of previous chapter. Particularly, we extend to a much more general estimator in three ways. First, in previous matrix group norm, each group is regularized by ℓ_2 norm. Now we extend it to any norm. Second, each group can be further restricted to any conic constraint. The positive semidefinite cone in the previous chapter is a special case. Last but not least, we consider any convex and differential function as our objective. The maximum likelihood of covariance matrix in the previous chapter should also be a special case in this chapter.

With more complicated constraints, one may expect the computation cost to grow much faster than linear in the dimension of the problem. However, we show that the same technique in previous chapter can be extended in a general form here. If we can assume any solution is a linear combination of a small set of "chunks", the cost can be reduced to an amount dependent on the size of each chunk times the number of chunks. As long as there is only a modest number of chunks, we can reduce the computation cost

per step, while also reducing overfitting in the final solution.

The content and organization of this chapter are as follows. Section 6.1 proposes a general group norm. Then the formal estimator with group structure is demonstrated in Section 6.2. A scalable optimization procedure based on Frank-Wolfe method is shown in Section 6.3. We present more model problems, such as sparse matrix nearness problem and sparse trace regression problem, in Section 6.4.

6.1 Generalized Group Norm

We consider the general vectorized form. Note that it is common to use vector notation for matrix variables. For example, given a matrix $X \in \mathbb{S}^{p\times p}$, one can map it to a vector x of length $p(p+1)/2$ containing the upper-triangular entries. For an index set $\gamma \subset \{1,\dots,p\}$ and a vector $x \in \mathbb{R}^p$, define x_γ as the subvector of x indexed by γ. We define the augmenting linear map $\gamma : \mathbb{R}^{|\gamma|} \to \mathbb{R}^p$ such that $(A_\gamma u)_\gamma = u,\ (A_\gamma u)_i = 0$ if $i \notin \gamma$.

We define our general form of norm $\|\cdot\|_A$ is the optimal value of

$$\begin{array}{ll} \underset{u_1,\dots,u_l}{\text{minimize}} & \sum_k w_k \|u_k\|_a \\ \text{subject to} & \sum_k A_{\gamma_k} u_k = x \end{array} \tag{6.1.1}$$

where $w_1,\dots,w_l$ are sets of positive weights and $\|\cdot\|_a$ could be any valid norm depending on different purposes. For example, it can be ℓ_p-norm($p \geq 1$) or ℓ_∞-norm for vector variables. It also can be Frobenius norm or nuclear norm for matrix variables. To show that $\|\cdot\|_A$ is a norm, the positivity and hold trivially. For the triangle inequality, let $x^{(1)} = \sum_k A_{\gamma_k} y_k$ and $x^{(2)} = \sum_k A_{\gamma_k} z_k$ be the optimal solutions in finding $\|x^{(1)}\|_A$ and $\|x^{(2)}\|_A$. Then $\sum_k A_{\gamma_k}(y_k + z_k) = x^{(1)} + x^{(2)}$. Additionally, by triangle inequality of $\|\cdot\|_a$, we have that $\sum_k w_k(\|y_k\|_a + \|z_k\|_a) \geq \sum_k w_k \|y_k + z_k\|_a$. Therefore $\|x^{(1)}\|_A + \|x^{(2)}\|_A \geq \|x^{(1)} + x^{(2)}\|_A$.

Next we show that the latent group matrix norm (5.2.3) in Section 5.2 of previous chapter is a special case. Recall $\|X\|_G$ is optimal solution of

$$\begin{array}{ll} \underset{v,U_1,\ldots,U_l}{\text{minimize}} & w_0\|v\|_2 + \sum_{k=1}^{l} w_k\|U_k\|_F \\ \text{subject to} & \mathbf{diag}(v) + \sum_{k=1}^{l} \mathcal{A}_{\beta_k}(U_k) = X. \end{array}$$

If one takes $\|\cdot\|_a$ for groups $u_k (k = 1, \cdots, l)$ as Frobenius norm and the other diagonal group $u_{l+1} = v$ as ℓ_2 norm, then $\|X\|_G$ is a special form of (6.1.1).

6.2 General Estimator with Group Structure

Our group structure estimator is generalized in three aspects. First, we generalize the group norm in last section. Second we consider any conic constraint. The positive semidefinite cone in previous chapter is a special case. Last but not least, we consider any convex and differential function as our objective.

The cone can cover many important examples, especially common for matrix variable as many applications requires the positive semidefiniteness. A set $\mathcal{C}$ is a cone if for every $x \in \mathcal{C}$ and $\theta \geq 0$, we have $\theta x \in \mathcal{C}$. A set $\mathcal{C}$ is a convex cone if for any $x_1, x_2 \in \mathcal{C}$ and $\theta_1, \theta_2 \geq 0$, we have $\theta_1 x_1 + \theta_2 x_2 \in \mathcal{C}$. A set $\mathcal{C}$ is a proper cone if $\mathcal{C}$ is convex, closed, solid (non-empty interior) and pointed (if $x \in \mathcal{C}$ and $-x \in \mathcal{C}$ then $x = 0$).

Our general group structure model in a vectorized optimization is formulated as follows.

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{s.t.} & \sum_{k=1}^{l} A_{\gamma_k} u_k = x \\ & \sum_{k=1}^{l} w_k\|u_k\|_a \leq \alpha \\ & u_k \in \mathcal{C}_k. \end{array} \tag{6.2.2}$$

Here the vector variable is $x \in \mathbb{R}^p$, $f(x)$ is any differentiable convex function, and

$\mathcal{C}_1, \ldots, \mathcal{C}_l$ are proper convex cones. The parameters $w_1, \ldots, w_l > 0$ are positive weights. α is the parameter to be tuned. $\gamma_1, \ldots, \gamma_l$ are indices for predefined groups. The augmenting binary matrices $A_\gamma \in \mathbb{R}^{p \times |\gamma|}$ are defined for subvector x_γ as

$$(A_\gamma x_\gamma)_i = \begin{cases} x_i, & i \in \gamma \\ 0, & \text{else.} \end{cases}$$

We remark that this is an approximate solution to $\{\min_x f(x) : \|x\|_A \leq \alpha, x \in \mathcal{C}\}$, where $\|\cdot\|_A$ is defined in (6.1.1). They can be equivalent as long as $\mathcal{C}$ is a partial separable cone defined as $\mathcal{C} = \{x \in \mathbb{R}^p \mid x_{\gamma_k} \in \mathcal{C}_k, k = 1, \cdots, l\}$. Essentially, many important cones are partial separable. In linear optimization, $\mathcal{C} = \mathbb{R}^p_+$ is a product of one-dimensional cones and thus separable. In semidefinite programming, $\mathcal{C} = \mathbb{S}^p_+$ is a partial separable cone and the equivalence has been discussed in Section 5.3. The implicit sparsity pattern of our estimator is constrained to the groups $\gamma_1, \cdots, \gamma_l$, i.e. $E(\{\gamma_k\}_1^l) = \cup_k \{i \mid i, \exists \gamma_k \supset i\}$.

6.3 Scalable Optimization Procedure

In last chapter, we apply Frank-Wolfe method to solve the structured sparse inverse covariance estimation and show its high efficiency. In this section, we show the same routine can be applied in a more general way.

Recall that the Frank-Wolfe method is designed for $\min_x \{f(x) : x \in \mathcal{D}\}$ where $\mathcal{D}$ is now defined as

$$\mathcal{D} = \left\{ x = \sum_{k=1}^{l} w_k A_{\gamma_k} u_k, \ \sum_{k=1}^{l} w_k \|u_k\|_a \leq \alpha, u_k \in \mathcal{C}_k \right\}. \tag{6.3.3}$$

To compute the forward step, we reformulate it after eliminating s to

$$\begin{aligned} \underset{u_k}{\text{minimize}} \quad & \langle \nabla f(x), \sum_{k=1}^{l} A_{\gamma_k} u_k \rangle \\ \text{s.t.} \quad & \sum_{k=1}^{l} w_k \|u_k\|_a \leq \alpha \\ & u_k \in \mathcal{C}_k. \end{aligned} \tag{6.3.4}$$

Proposition 6.3.1 *The solution to* (6.3.4) *on problem* (6.2.2) *is*

$$u_j^* = \frac{\alpha}{w_j \|z_j\|_{a^*}} z_j, \quad u_k = 0, \ \forall k \neq j.$$

for $j = \arg\max_k w_k^{-1} \|z_k\|_{a^*}$ *and* $z_k = \mathbf{proj}_{\mathcal{C}_j}(-\nabla f(x)_{\gamma_k})$*, where* $\|\cdot\|_{a^*}$ *denotes the dual norm of* $\|\cdot\|_a$*.*

Proof.

For notational convenience, define $c_k = \nabla f(x)_{\gamma_k}$ for $k = 1, \ldots, l$. Then we can rewrite the objective function in (6.3.4) as

$$\underset{u_k}{\text{maximize}} \sum_k (-c_k)^T u_k$$

with constraints unchanged. From Moreau's decomposition, any vector a can be written as the sum of its projection on a closed convex cone $\mathcal{C}$ and its polar cone $\mathcal{C}^\circ$, of which are orthogonal. If we expand

$$(-c_k)^T u_k = \mathbf{proj}_{\mathcal{C}_k}(-c_k)^T u_k + \mathbf{proj}_{\mathcal{C}_k^\circ}(-c_k)^T u_k$$

then since feasible $u_k \in \mathcal{C}_k$, by definition of polar cone $\mathbf{proj}_{\mathcal{C}_k^\circ}(-c_k)^T u_k \leq 0$, and $= 0$ only if $u_k = v_k \mathbf{proj}_{\mathcal{C}_k}(-c_k)$. This is the optimal choice of direction for u_k, since it also

maximizes the first term $\mathbf{proj}_{\mathcal{C}_k}(-c_k)^T u_k$, and does not affect the norm constraint. We then can simplify and rewrite (6.3.4) without the conic constraint as the negative of

$$\begin{array}{ll} \underset{u_k}{\text{maximize}} & \sum_{k=1}^{l} \mathbf{proj}_{\mathcal{C}_k}(-c_k)^T u_k \\ \text{s.t.} & \sum_{k=1}^{l} w_k \|u_k\|_a \leq \alpha. \end{array} \tag{6.3.5}$$

Note that the constraint $u_k \in \mathcal{C}_k$ is now implicitly enforced, since any component not in $\mathcal{C}_k$ will not help maximize the objective but will push against the bound. Similar to Lemma 3 and proof in [90], we further reformulate it as follows.

$$\begin{array}{ll} \underset{u_k,\eta_k}{\text{maximize}} & \sum_{k=1}^{l} \mathbf{proj}_{\mathcal{C}_k}(-c_k)^T u_k \\ \text{s.t.} & \sum_{k=1}^{l} \eta_k \leq 1 \\ & {}^{w_k}\!/_{\alpha} \|u_k\|_a \leq \eta_k, \forall k = 1, \cdots, l. \end{array} \tag{6.3.6}$$

We can solve the maximization over u_k first and then over η_k. Once η_k is fixed, maximizing u_k can be done separately for each group k and the subproblem is derived from the definition of dual norm, which results in

$$\begin{array}{ll} \underset{\eta_k}{\text{maximize}} & \sum_{k=1}^{l} \eta_k \frac{\alpha}{w_k} \|\mathbf{proj}_{\mathcal{C}_k}(-c_k)^T\|_{a^*} \\ \text{s.t.} & \sum_{k=1}^{l} \eta_k \leq 1. \end{array}$$

The final objective value is then given by $\alpha \max_k w_k^{-1} \|\mathbf{proj}_{\mathcal{C}_k}(-c_k)\|_{a^*}$. Defining $j = \arg\max_k w_k^{-1} \|\mathbf{proj}_{\mathcal{C}_k}(-c_k)\|_{a^*}$, we see that this is satisfied if $u_k = 0$ for all $k \neq j$, and$u_j = \frac{\alpha}{w_j \|\mathbf{proj}_{\mathcal{C}_j}(-c_j)\|_{a^*}} \cdot \mathbf{proj}_{\mathcal{C}_j}(-c_j)$ □

We summarize the entire procedure in Algorithm 18. A significant computational challenge is the enforcement of conic constraint. If the variable is matrix $X \in \mathbb{S}^p$ and

Algorithm 18 One step of Frank-Wolfe algorithm for (6.2.2)

Input: $x^{[t]} \in \mathcal{D}$ and step size $\eta^{[t]} = \frac{2}{t+2}$ at t-th iteration, the norm a.

1: Find $\nabla f(x)$.

2: Find the forward direction u^+:

$$z_k = \mathbf{proj}_{\mathcal{C}_j}(-\nabla f(x)_{\gamma_k}).$$
$$j = \arg\max_k w_k^{-1} \|z_k\|_{a^*}.$$
$$u_j^+ = \frac{\alpha}{w_j \|z_j\|_{a^*}} z_j, \quad u_k^+ = 0, \ \forall k \neq j.$$

3: Update $x^{[t+1]} = x^{[t]} + \eta^{[t]} u^+$.

Output: $x^{[t+1]}$.

conic constraint is $X \succeq 0$, interior point methods require solving a KKT system which usually is expensive. And even gradient descent and block coordinate type methods may require a sequence of eigenvalue decompositions $O(p^3)$. Since we have enforced sparsity, we may consider using sparse eigenvalue methods (like Lanczos type methods) but in practice these are also slow, since we usually need to compute a full spectrum (and not just a top or bottom eigenvalue). Without further decomposition, it will not be possible to solve PSD constraint for large p without memory limitations.

As in Algorithm 6.2.2, the per-iteration computational complexity lies in the forward step (line 2). As discussed in [68], if the unit ball of the atomic norm is polyhedral, then each forward step with high probability is a vertex of the norm ball. If this is the case, then the forward step reduces to finding the atom most correlated with the gradient at each step. Unfortunately, the feasibility set of (6.2.2) is not polyhedral; specifically, the cones $\mathcal{C}_k$ will not have “straight walls".

Nevertheless, the complexity of our algorithm is still significantly reduced. The forward step is simply a sequence of projections on small cones. Note that this is not exactly a one-step procedure, because we must compute l projections before picking the best atom. However, in our case computing l projections on $|\gamma_k|$-dimensional cones is much cheaper than projecting on the whole p-dimensional cone. Moreover, the projection for each group in one iteration can be computed independently. Hence Algorithm 18 is

desired in a distributed computing setting and can further be accelerated.

6.4 Examples under the Framework

We first show the structured inverse covariance estimator (5.3.8) in Section 5.2 and the optimization procedure Algorithm 16 in Section 5.4 of last chapter is a special case of (6.2.2) and Algorithm 18. We then present more examples under the framework of our general estimator and optimization procedure, such as sparse matrix nearness problem and sparse trace regression problem.

Inverse Covariance Estimation

To show inverse covariance estimation (5.3.6) is a special case of (6.2.2), let $m = p^2$, and objective function be

$$f(\mathbf{vec}(X)) = -\text{logdet}(X) + \text{tr}\,(\hat{C}X). \tag{6.4.7}$$

with constraint $\mathcal{C}_k = \mathbb{S}_+^{|\beta_k|}, k = 1, \ldots, l,$ (positive semi-definite cone) and index sets γ_k and β_k defined as

$$\mathbf{vec}(X)_{\gamma_k} = \mathbf{vec}(X_{\beta_k,\beta_k}),\ k = 1, \ldots, l.$$

By setting the norm $\|\cdot\|_a$ in (6.2.2) as the Frobenius norm $\|\cdot\|_F$, then formulations (5.3.8) and (6.2.2) are equivalent. In the optimization, Frobenius norm is self-dual so that resulting $\|\cdot\|_{a^*}$ is still Frobenius norm $\|\cdot\|_F$.

Sparse Matrix Nearness Problem

The matrix nearness problem essentially projects a given matrix on the set of matrices that satisfy certain property. Problems of this type and its variant has great popularity in applications such as finance, computational biology and sensor networks [61, 97, 111]. Particularly, given an arbitrary matrix C, the goal of matrix nearness problem is to find

an approximation matrix X that possess some structural properties $\mathcal{S}$. Mathematically, it could be represented as

$$\underset{X}{\text{minimize}} \quad f(X) = \tfrac{1}{2}\|X - C\|^2 \qquad \text{s.t.} \quad X \in \mathcal{S}. \tag{6.4.8}$$

We can specify the distance between X and C through the norm of their difference. The choice of the norms are usually determined by the tractability of the problem. The most two common norms are Frobenius norm $\|\cdot\|_F$ and matrix 2-norm $\|\cdot\|_2$.

In terms of the structure S, the positive semidefinite constraint is widely studied and in principle can be solved by standard solvers such as `CVX` [54] or `MOSAK` [3]. However, the computation involves inverse of dense matrix and not scalable with large dimension size p.

The sparse matrix nearness problem, on the other hand, restricts the approximation X with additional structural constraint E. The exact sparsity pattern should depend on the different applications and needs. For example, if the given matrix C is a sparse covariance matrix or inverse covariance matrix, then the sparsity pattern corresponds to the dependencies of variables. Specifically for the group structure we consider in Section 6.1 for the matrix variable, the problem and the sparsity pattern are characterized as

$$\underset{X}{\text{minimize}} \quad f(X) = \tfrac{1}{2}\|X - C\|_F^2 \qquad \text{s.t.} \quad X \in \mathbb{S}_+^p \cap \mathbb{S}_E^p \tag{6.4.9}$$

where

$$E(\{\beta_k\}_1^l) = \{(i,j) \mid i \neq j, \exists \beta_k \supset \{i,j\}\}.$$

Here we use the distance as Frobenius norm. The objective function is convex and differentiable. The variable X is positive semidefinite and consists of a sequence of smaller groups indexed as β_k. This problem falls into the category of our estimator

(6.2.2) and can be efficiently solved by Algorithm 18.

Sparse Trace Regression Problem

The trace regression problem is of the form

$$b_i = \mathrm{tr}\,(C_i^T X) + \epsilon_i \quad \text{for} \quad i = 1, \cdots, n$$

where C_i is the given measurement matrix, b_i is the observations. The goal is to find the positive semidefinite matrix $X \in \mathbb{S}_+^p$ to fit the regression model. This model is of great interest in the "few samples, high dimension" regime in which sample size $n \ll p$. Trace regression has a wide range of applications including compressive sensing, matrix completion, phase retrieval and so on [17, 18, 20, 87]. A typical way to formulate the problem is as follows.

$$\underset{X}{\text{minimize}} \quad f(X) = 1/2 \sum_{i=1}^{n} (\mathrm{tr}\,(C_i^T X) - b_i)^2 \qquad \text{s.t.} \quad X \in \mathbb{S}_+^p. \tag{6.4.10}$$

Since the number of samples is smaller compared to dimension size in the setting, structural regularizations are commonly used to restrict the "search space" of the variables and obtain statistically stable result. There are two common regularizations, the matrix $\|\cdot\|_1$ norm for sparsity structure, and the nuclear norm $\|\cdot\|_*$ norm for low-rank structure [17, 87, 98, 104].

We consider structured sparse trace regression, in which prior information of group structure is known. Given the sparsity pattern $E(\{\beta_k\}_1^l) = \{(i,j) \mid i \neq j, \exists \beta_k \supset \{i,j\}\}$, the result is to find the solution for the following problem.

$$\underset{X}{\text{minimize}} \quad f(X) = 1/2 \sum_{i=1}^{n} (\mathrm{tr}\,(C_i X) - b_i)^2 \qquad \text{s.t.} \quad X \in \mathbb{S}_+^p \cap \mathbb{S}_E^p. \tag{6.4.11}$$

The structural regularization $\mathbb{S}_E^p$ implicitly imposes group sparsity indexed as β_k.

The objective function is quadratic. Hence problem (6.4.11) is another special case under the framework of our estimator (6.2.2) and can be efficiently solved by Algorithm 18.

Chapter 7

Tensor Low-Rank Decomposition and Completion

In previous chapters, we discuss the structures of sparsity and group sparsity. We use the model LASSO problem to analyze a wide class of scalable optimization algorithms. We also use the model Sparse Inverse Covariance Estimation problem to develop a group structured estimator and scalable Frank-Wolfe optimization method. A more general estimator is then proposed in the last chapter and a wide class of structured problems can be covered.

In this chapter, we conduct research on structures in the tensor problem. Tensor is a multidimensional array, and is a natural extension of vector and matrix. It can be factorized into the product of matrices or vectors, known as tensor decomposition. There are multiple ways to decompose a tensor. Two most common ways are called CP (short for CANDECOMP/PARAFAC) decomposition and Tucker decomposition. Both decompositions have a wide range of applications in signal processing, computer vision, data mining, graphical models, and so on. Tensor decomposition can also be understood as a generalization of matrix factorization. In Section 1.2.3 and Section 2.2, we give an

introduction of low-rank matrix decomposition, and show its applications in recommender systems. Though matrix factorization with (user, item) has been successful in many settings, it is also shown that more dimensions of variables can help better capture user's preference. In this chapter, we explore the low-rank structure in tensor decomposition.

In particular, we construct a regularized multiconvex formulation for tensor decomposition with the low CP and Tucker rank. The key underlying idea of getting low-rank tensor is to take advantage of group sparsity structure. We try $\ell_{1,2}$-norm regularization and the $\ell_{1,\infty}$-norm regularization to achieve the group sparse effect. By applying the Block Coordinate Descent method, the resulting subproblem is also a regularized convex optimization and can be efficiently solved by proximal method. As a natural application, we show that our approach can be applied to solve the tensor completion problem.

The rest of the chapter is organized as follows. Section 7.1 reviews the basic knowledge and notations of tensor. Section 7.2 describes the relationship between the low-rank decomposition of a tensor and the group sparsity property of its factor matrices. We formulate the problem and solve it by scalable method in Section 7.3. Section 7.4 is devoted to tensor low-rank completion. In Section 7.5, we provide some numerical results to justify our algorithms.

7.1 Review of Tensor Properties

Here we introduce the notations and properties we use throughout this thesis. We refer readers to a survey paper [73] and references therein for more detailed properties on tensor.

7.1.1 Notations

The order N of a tensor is the number of dimensions, also known as ways. The Frobenius norm of a N-th order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, by notation $\|\cdot\|$, is the square root of

the sum of the squares of all its elements, i.e.,

$$\|\mathcal{X}\| = \sqrt{\sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\cdots\sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N}^2}.$$

The mode-n matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $X_{(n)}$ and arranges the mode-n fibers to be the columns of the resulting matrix. Tensor element $(i_1, i_2, \cdots, i_N)$ maps to matrix element (i_n, j):

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{N} (i_k - 1) J_k \qquad \text{with} \qquad J_k = \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m.$$

We call tensor $\mathcal{X}$ is rank-one, if it is the outer product of some vectors. And the outer product of vectors $\mathbf{a}^1, \mathbf{a}^2, \cdots, \mathbf{a}^N$ with $\mathbf{a}^n \in \mathbb{R}^{I_n}$ is denoted by $\mathbf{a}^1 \circ \mathbf{a}^2 \circ \cdots \mathbf{a}^N$ such that

$$(\mathbf{a}^1 \circ \mathbf{a}^2 \circ \cdots \circ \mathbf{a}^N)_{i_1 i_2 \cdots i_N} = \mathbf{a}^1_{i_1} \mathbf{a}^2_{i_2} \cdots \mathbf{a}^N_{i_N}.$$

The n-mode (matrix) product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $U \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{X} \times_n U$ and is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$. Elementwise, we have

$$(\mathcal{X} \times_n U)_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} u_{j i_n}.$$

The symbol "$\otimes$" denotes the standard Kronecker product of matrices. For $A \in \mathbb{R}^{I \times K}$ and $B \in \mathbb{R}^{J \times K}$, the result is a matrix of size $(IK) \times (JL)$ defined by

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1J}B \\ a_{21}B & a_{22}B & \cdots & a_{2J}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}B & a_{I2}B & \cdots & a_{IJ}B \end{bmatrix}.$$

The Khatri-Rao product of matrices $A \in \mathbb{R}^{I\times K}$ and $B \in \mathbb{R}^{J\times K}$, is denoted by $A \odot B$. The result is a matrix of size $(IJ) \times (K)$ defined by

$$A \odot B = [\mathbf{a}_1 \otimes \mathbf{b}_1 \ \mathbf{a}_2 \otimes \mathbf{b}_2 \cdots \mathbf{a}_K \otimes \mathbf{b}_K].$$

7.1.2 CP Decomposition and CP Rank

The CP decomposition is to factorize a tensor into a sum of component rank-one tensors. In general, for N-way tensor, one can get the formulation of CP decomposition: $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$,

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}^{1,r} \circ \mathbf{a}^{2,r} \circ \cdots \circ \mathbf{a}^{N,r} := [\![A_1, A_2, \cdots, A_N]\!] \tag{7.1.1}$$

where "$\circ$" denotes the outer product of vectors and $A_n = [\mathbf{a}^{n,1}, \mathbf{a}^{n,2}, \cdots, \mathbf{a}^{n,R}] \in \mathbb{R}^{I_n\times R}$ for $1 \le n \le N$ are factor matrices. Moreover, CP decomposition can be expressed in the matricized version. The mode-n matricized CP decomposition is

$$X_{(n)} = A_n(A_N \odot \cdots \odot A_{n+1} \odot A_{n-1} \odot \cdots \odot A_1)^{\mathrm{T}}. \tag{7.1.2}$$

The CP rank for a tensor $\mathcal{X}$, denoted by $\mathbf{rank}_{CP}$, is a natural generalization of the rank for matrix. It is associated with the smallest CP decomposition defined as follows.

$$\mathbf{rank}_{CP}(\mathcal{X}) = \min\left\{ R \;\middle|\; \mathcal{X} = \sum_{r=1}^{R} a^{1,r} \circ a^{2,r} \circ \cdots \circ a^{N,r} \right\} \tag{7.1.3}$$

In fact, to find the smallest CP decomposition of a given tensor is very challenging, known to be NP-complete [58]. Even for a specific $9 \times 9 \times 9$ tensor, we only know its CP rank lies in between 18 and 23 [75]. Hence research on low-rank decomposition becomes a big motivation.

7.1.3 Tucker Decomposition and Tucker Rank

The Tucker decomposition is to decompose a tensor in to a core tensor multiplied by a matrix along each mode. In general, for N-way tensors, the formulation of Tucker decomposition is

$$\begin{aligned} \mathcal{X} &= \mathcal{G} \times_1 A_1 \times_2 A_2 \cdots \times_N A_N \\ &= \sum_{i_1=1}^{r_1} \sum_{i_2=1}^{r_2} \cdots \sum_{i_N=1}^{r_N} \mathcal{G}_{i_1 i_2 \cdots i_N} \mathbf{a}^{1,i_1} \circ \mathbf{a}^{2,i_2} \circ \cdots \circ \mathbf{a}^{N,i_N} \\ &:= [\![\mathcal{G}; A_1, A_2, \cdots, A_N]\!] \end{aligned} \tag{7.1.4}$$

where $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_N}$ is the so-called core tensor, $A_n = [\mathbf{a}^{n,1}, \mathbf{a}^{n,2}, \cdots, \mathbf{a}^{n,r_n}] \in \mathbb{R}^{I_n \times r_n}$ for $1 \leq n \leq N$ are factor matrices. Moreover, Tucker decomposition can be expressed in the matricized version. The mode-n matricized Tucker decomposition is

$$X_{(n)} = A_n G_{(n)} (A_N \otimes \cdots \otimes A_{n+1} \otimes A_{n-1} \otimes \cdots \otimes A_1)^{\mathrm{T}}. \tag{7.1.5}$$

The Tucker rank of a tensor denoted by $\mathbf{rank}_{TK}$ is a vector corresponding to the size of the core tensor associated with the *smallest* Tucker decomposition. Typically, a Tucker rank $(r_1, r_2, \cdots, r_N)$ means that the size of the core tensor is $r_1 \times r_2 \times \cdots \times r_N$. It has been shown in the literature [73] that, if $A_1, \cdots, A_N$ are all orthogonal, the smallest Tucker decomposition can be accomplished in polynomial time.

7.2 Group Structural Constraint for Low-Rank Tensor

The typical tensor decomposition was proposed in [22] and [71], in which the rank of tensor is assumed to be a prerequisite knowledge. However, as mentioned in last section, the rank of tensor is difficult to compute. Hence we develop a method for computing low-rank approximation without prior knowledge regarding the rank of the tensor. Essentially, we

formulate a multi-block optimization problem. The objective function consists of two parts. One is the least square term for the traditional tensor decomposition, while the other one is the group structural regularization for low-rank effect.

In this section, we construct the group structural constraint to achieve the low-rank effect. We show that obtaining the low-rank decomposition of a tensor can be transformed into the problem of finding the factor matrices with as many zero columns as possible. The group structure is motivated by the following key observations.

Proposition 7.2.1 *In terms of the matricized version of decomposition, we have the following:*

(i) In CP decomposition (7.1.1)*, suppose there is a zero column of matrix A_n for some $1 \leq n \leq N$, then this decomposition actually admits at most $R-1$ rank-one terms. Therefore, obtaining a low-rank CP decomposition can be achieved by generating a decomposition with as many zero column vectors as possible in either $A_1, A_2, \cdots, A_N$.*

(ii) In Tucker decompostion (7.1.4)*, suppose there is a zero column of matrix A_n for some $1 \leq n \leq N$, then this decomposition actually admits a core with size at most $(r_1, \cdots, r_{n-1}, r_n - 1, r_{n+1}, \cdots, _N)$. Therefore, obtaining a low-rank Tucker decomposition can be achieved by generatinga decomposition with all the matrices $A_1, A_2, \cdots, A_N$ have as many zero columns as possible.*

Proof. (i) In CP decomposition (7.1.1), we assume that the j-th column of A_n is a zero vector for some $1 \leq j \leq R$ and $1 \leq n \leq N$ in CP decomposition (7.1.1), namely $a^{n,j} = 0$. Then

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}^{1,r} \circ \mathbf{a}^{2,r} \circ \cdots \circ \mathbf{a}^{N,r} = \sum_{r=1, r \neq j}^{R} \mathbf{a}^{1,r} \circ \mathbf{a}^{2,r} \circ \cdots \circ \mathbf{a}^{N,r},$$

where the number of rank-one terms is at most $R-1$. Therefore, if there are many zero columns in either $A_1, A_2, \cdots, A_N$, then $\mathcal{X}$ is of low CP rank.

(ii) In Tucker decomposition (7.1.4), without loss of generality, we assume $\mathbf{a}^{1,1} = 0$,

which means the first column of A_1 is a zero vector. Then

$$\begin{aligned}\mathcal{X} &= \sum_{i_1=1}^{r_1}\sum_{i_2=1}^{r_2}\cdots\sum_{i_N=1}^{r_N}\mathcal{G}_{i_1 i_2\cdots i_N}\mathbf{a}^{1,i_1}\circ\mathbf{a}^{2,i_2}\circ\cdots\circ\mathbf{a}^{N,i_N}\\ &= \sum_{i_1=2}^{r_1}\sum_{i_2=1}^{r_2}\cdots\sum_{i_N=1}^{r_N}\mathcal{G}_{i_1 i_2\cdots i_N}\mathbf{a}^{1,i_1}\circ\mathbf{a}^{2,i_2}\circ\cdots\circ\mathbf{a}^{N,i_N}.\end{aligned}$$

By constructing $\hat{\mathcal{G}}\in\mathbb{R}^{(r_1-1)\times r_2\times\cdots\times r_N}$ and $\hat{A}_1\in\mathbb{R}^{I_1\times(r_1-1)}$ such that

$$\hat{\mathcal{G}}_{i_1,i_2,\cdots,i_N}=\mathcal{G}_{i_1+1,i_2,\cdots,i_N}\quad\text{and}\quad(\hat{A}_1)_{j_1,j_2}=(A_1)_{j_1+1,j_2},$$

we have that $[\hat{\mathcal{G}};\hat{A}_1,\cdots,A_N]$ is a valid Tucker decomposition of $\mathcal{X}$ with core size $(r_1-1,r_2,\cdots,r_N)$. Therefore, if all $A_1,A_2,\cdots,A_N$ have many zero column vectors, then $\mathcal{X}$ is of low Tucker rank. □

Proposition 7.2.1 implies low tensor rank corresponds to the group sparsity of $A_1,A_2,\cdots,A_N$. If we treat columns of factor matrix as groups with each group all zeros, we can obtain the low-rank tensor. To take advantage of this observation, we write low-rank decomposition explicitly in terms of matrices $A_1,A_2,\cdots,A_N$, and we use the matricized version (7.1.2), (7.1.5). As discussed in the Section 2.1 and 1.2.2, ℓ_1-norm and $\ell_{1,2}$-norm can lead to sparsity and group sparsity. In our model, we treat each column as a group and apply $\ell_{1,2}$-norm regularization.

$$\ell_{1,2}\text{ based low-rank structural constraint}=\left(\sum_{t=1}^{R}\sum_{i=1}^{N}\|\mathbf{a}^{i,t}\|_2\right)$$

where $\mathbf{a}^{i,t}$ is the t-th column of matrix A_i for $1\le i\le N$ and $1\le t\le R$. Besides the $\ell_{1,2}$-norm regularization, there is a natural consideration arises from the question about whether the $\ell_{1,2}$-norm can be replaced by some other $\ell_{1,p}$-norms for which $p>2$. Since it is the ℓ_1-norm that really promotes the sparsity structure, the choice of the ℓ_2-norm in the group LASSO might not be necessarily unchangeable. In practice, another popular

choice is the $\ell_{1,\infty}$-norm regularization [42].

$$\ell_{1,\infty} \text{ based low-rank structural constraint} = \left(\sum_{t=1}^{R} \sum_{i=1}^{N} \|\mathbf{a}^{i,t}\|_{\infty} \right)$$

Consequently, the problems of tensor low-rank decomposition and the completion can both be casted as matrix optimization problems.

7.3 Tensor Low-Rank Decomposition

In this section, we present the idea and formulation to find low-rank approximation of a N-way tensor. Particularly, suppose we are given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and we do not know its exact rank, what we would do is to start with a relatively large rank and implement our new approach of group sparsity to decompose $\mathcal{X}$. We illustrate in detail on CP low-rank decomposition in this section. Tucker low-rank decomposition can be done in a similar routine and hence omitted. We remark that Tucker decomposition can be accomplished in polynomial time in the literature.

7.3.1 Formulation as Block Regularized Optimization

In CP low-rank approximation, the goal is to factorize a tensor into a sum of rank-one tensors with as less terms as possible and approximates the original tensor well. Our formulation is

$$\min_{A_1, A_2, \cdots, A_N} \frac{1}{2} \|\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!]\|^2 + \rho \left(\sum_{t=1}^{R} \sum_{i=1}^{N} \|\mathbf{a}^{i,t}\|_2 \right) \tag{7.3.6}$$

and

$$\min_{A_1, A_2, \cdots, A_N} \frac{1}{2} \|\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!]\|^2 + \rho \left(\sum_{t=1}^{R} \sum_{i=1}^{N} \|\mathbf{a}^{i,t}\|_{\infty} \right) \tag{7.3.7}$$

for different structural constraints. ρ is the parameter to control the degree, $\mathbf{a}^{i,t}$ is the t-th column of matrix A_i for $1 \leq i \leq N$ and $1 \leq t \leq R$. The objective in either (7.3.6) or (7.3.7) are in the framework of the following regularized multiconvex optimization.

$$\min_{x} f(x_1, \cdots, x_N) + \sum_{i=1}^{N} r_i(x_i), \tag{7.3.8}$$

where $x_1, \cdots, x_N$ is the decision variables, f is assumed to be a differentiable and block multiconvex function, that is f is a convex function of x_i while all the other blocks are fixed, and $r_i, i = 1, \cdots, N$ are convex functions. With regard to our formulation, it is

$$\min_{A_1, A_2, \cdots, A_N} f(A_1, A_2, \cdots, A_N) + \sum_{i=1}^{N} r_i(A_i). \tag{7.3.9}$$

where

$$r_i(A_i) = \rho \sum_{t=1}^{R} \|a^{i,t}\|_p$$

for $1 \leq i \leq N$, $p \in \{2, \infty\}$ is convex with respect to A_i and

$$f(A_1, A_2, \cdots, A_N) = \frac{1}{2}\|X_{(i)} - A_i(A_N \odot \cdots A_{i+1} \odot A_{i-1} \cdots A_1)^{\mathrm{T}}\|^2$$

implying that $f(A_1, A_2, \cdots, A_N)$ is a differentiable and block multiconvex function. Moreover, the i-th block-partial gradient of f is given by

$$\nabla_{A_i} f = (A_i(A_N \odot \cdots A_{i+1} \odot A_{i-1} \cdots A_1)^{\mathrm{T}} - X_{(i)})(A_N \odot \cdots A_{i+1} \odot A_{i-1} \cdots A_1) \tag{7.3.10}$$

and the partial gradient function $\nabla_{A_i} f$ is Lipschitz continuous. In conclusion, the problem (7.3.9) can fit into the regularized multiconvex optimization framework in [126] and the algorithms therein can be applied.

7.3.2 Block Coordinate Descent with Prox-linear Method

For multiblock optimization problem (7.3.9), we apply block coordinate descent of Gauss-Seidel type. Essentially, it cyclically updates each of $A_1, \cdots, A_N$ while fixing the remaining blocks at their last updated values. To be more specific, in the k-th iteration, updating scheme of BCD is described as follows:

$$A_i^{[k]} = \underset{A_i}{\operatorname{argmin}} F_i(A_1^{[k]}, \cdots, A_{i-1}^{[k]}, A_i, A_{i+1}^{[k-1]}, \cdots, A_N^{[k-1]}), \tag{7.3.11}$$

where F_i is certain multiblock function for $1 \leq i \leq N$. For example, one natural choice of F_i is

$$\begin{aligned} & F_i(A_1^{[k]}, \cdots, A_{i-1}^{[k]}, A_i, A_{i+1}^{[k-1]}, \cdots, A_N^{[k-1]}) \\ = \quad & f(A_1^{[k]}, \cdots, A_{i-1}^{[k]}, A_i, A_{i+1}^{[k-1]}, \cdots, A_N^{[k-1]}) + r_i(A_i). \end{aligned}$$

In [126], three alternative choices of F_i were provided under the scheme of BCD, and the authors proved that the sequence generated by BCD converges to critical point of (7.3.9) under certain mild conditions. For practical purpose, we adopt the prox-linear calculating rule in [126] and have the following regularized convex problem:

$$A_i^{[k]} = \underset{A_i}{\operatorname{argmin}} \langle H_i^{[k]}, A_i - \hat{A}_i^{[k-1]} \rangle + \frac{L_i^{[k-1]}}{2} \|A_i - \hat{A}_i^{[k-1]}\|^2 + \rho \sum_{t=1}^{R} \|a^{i,t}\|_p \tag{7.3.12}$$

where $p \in \{2, \infty\}$, $H_i^{[k]} = \nabla f_{A_i}(A_1^{[k]}, \cdots, \hat{A}_i^{[k-1]}, A_{i+1}^{[k-1]}, \cdots, A_N^{[k-1]})$ is the block-partial gradient of f at $\hat{A}_i^{[k-1]}$ and $\hat{A}_i^{[k-1]}$ is given by the following formula:

$$\hat{A}_i^{[k-1]} = A_i^{[k-1]} + \omega_i^{[k-1]}(A_i^{[k-1]} - A_i^{[k-2]})$$

with $\omega_i^{[k-1]} \geq 0$ being the extrapolation weight. It is reported in [126] that this particular calculating rule appears to be very efficient in many tests.

Specify the Parameters

In the following, we illustrate how to set the parameters $L_i^{[k-1]}$ and $\omega_i^{[k-1]}$ in order to satisfy the conditions in [126]. For the i-th block, let

$$P_i^{[k-1]} = A_N^{[k-1]} \odot \cdots A_{i+1}^{[k-1]} \odot A_{i-1}^{[k]} \cdots A_1^{[k]}, \tag{7.3.13}$$

$$L_i^{[k-1]} = \max\left\{ l^{k-2}, \|(P_i^{[k-1]})^{\mathrm{T}} P_i^{[k-1]}\| \right\} \text{ for } 1 \le i \le N, \tag{7.3.14}$$

where $l^{k-2} = \min_{1\le i\le N} L_i^{[k-2]}$.

We take $t_0 = 1$, $t_k = \frac{1}{2}\left(1 + \sqrt{(1 + 4t_{k-1}^2)}\right)$ and $\hat{\omega}_{k-1} = \frac{t_{k-1}-1}{t_k}$; then

$$\omega_i^{[k-1]} = \min\left(\hat{\omega}_{k-1}, \delta_\omega \sqrt{\frac{l^{k-2}}{L_i^{[k-1]}}}\right) \tag{7.3.15}$$

where $\delta_\omega < 1$ is pre-selected. Therefore, under the current notation, $\hat{A}_i^{[k-1]} = A_i^{[k-1]} + \omega_i^{[k-1]}(A_i^{[k-1]} - A_i^{[k-2]})$ and

$$H_i^{[k]} = \left(\hat{A}_i^{[k-1]}(P_i^{[k-1]})^{\mathrm{T}} - X_{(i)}\right) P_i^{[k-1]}, \quad \forall 1 \le i \le N. \tag{7.3.16}$$

Closed Form Solution for the Subproblem

Now suppose $H_i^{[k]}$, $L_i^{[k-1]}$ and $\hat{A}_i^{[k-1]}$ are given, let's investigate how to solve (7.3.12). We first consider the case when $p = 2$. Notice that this problem is separable in columns, so it can be transformed as

$$\begin{aligned}
(\mathbf{a}^t)^{[k]} &= \underset{\mathbf{a}^t}{\operatorname{argmin}} \langle (\mathbf{h}^t)^{[k]}, \mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]} \rangle + \frac{L_i^{[k-1]}}{2} \|\mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]}\|^2 + \rho \|\mathbf{a}^t\|_2 \\
&= \underset{a^t}{\operatorname{argmin}} \frac{L_i^{[k-1]}}{2} \left\| \mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]} + \frac{(\mathbf{h}^t)^{[k]}}{L_i^{[k-1]}} \right\|^2 + \rho \|\mathbf{a}^t\|_2
\end{aligned}$$

where $\mathbf{a}^t, (\mathbf{a}^t)^{[k]}, (\hat{\mathbf{a}}^t)^{[k-1]}, (\mathbf{h}^t)^{[k]}$ are the t-th column of $A_i, A_i^{[k]}, \hat{A}_i^{[k-1]}, H_i^{[k]}$ respectively.

This update can be calculated explicitly by using *soft-thresholding operator* [36]:

$$(\mathbf{a}^t)^{[k]} = \max\{\|(\mathbf{b}^t)^{[k]}\|_2 - \rho, 0\}\frac{(\mathbf{b}^t)^{[k]}}{L_i^{[k-1]}\|(\mathbf{b}^t)^{[k]}\|_2}, \tag{7.3.17}$$

where

$$(\mathbf{b}^t)^{[k]} = L_i^{[k-1]}(\hat{\mathbf{a}}^t)^{[k-1]} - (\mathbf{h}^t)^{[k]}.$$

Now let's investigate how to solve (7.3.12) when $p = \infty$. Similar to the discussion above, suppose $\mathbf{a}^t, (\mathbf{a}^t)^{[k]}, (\hat{\mathbf{a}}^t)^{[k-1]}, (\mathbf{h}^t)^{[k]}$ are the t-th column of $A_i, A_i^{[k]}, \hat{A}_i^{[k-1]}, H_i^{[k]}$ respectively, and the subproblem can be transformed as

$$\begin{aligned}(\mathbf{a}^t)^{[k]} &= \underset{\mathbf{a}^t}{\operatorname{argmin}}\langle(\mathbf{h}^t)^{[k]}, \mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]}\rangle + \frac{L_i^{[k-1]}}{2}\|\mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]}\|^2 + \rho\|\mathbf{a}^t\|_\infty \\ &= \underset{\mathbf{a}^t}{\operatorname{argmin}}\frac{L_i^{[k-1]}}{2}\left\|\mathbf{a}^t - (\hat{\mathbf{a}}^t)^{[k-1]} + \frac{(\mathbf{h}^t)^{[k]}}{L_i^{[k-1]}}\right\|^2 + \rho\|\mathbf{a}^t\|_\infty. \end{aligned}\tag{7.3.18}$$

This update can be performed by resorting to the following lemma.

Lemma 7.3.1 *Suppose* $\mathbf{a}^* = \underset{\mathbf{a}\in\mathbb{R}^m}{\operatorname{argmin}}\frac{1}{2}\|\mathbf{a} - \mathbf{b}\|^2 + \gamma\|\mathbf{a}\|_\infty$, *then we have*

$$a_i^* = \begin{cases} b_i & \text{if} \quad |b_i| \le y^* \\ y^* & \text{if} \quad |b_i| > y^* \end{cases} \tag{7.3.19}$$

for $i = 1, 2, \ldots, m$, *where* $y^* = \underset{y\ge 0}{\operatorname{argmin}}\frac{1}{2}\sum_{i=1}^{m}[(|b_i| - y)_+]^2 + \gamma y$, *and* $(x)_+ = \max(x, 0)$.

Proof. We first observe that if $\|\mathbf{a}\|_\infty$ is determined say $\|\mathbf{a}\|_\infty = y$, to minimize $\|\mathbf{a} - \mathbf{b}\|^2$ we can choose a_i (the ith component of vector $\mathbf{a}$) such that the following holds

$$\|\mathbf{a} - \mathbf{b}\|^2 = \sum_{i=1}^{m}[(|b_i| - y)_+]^2. \tag{7.3.20}$$

Therefore, the target problem can be further converted to the following,

$$\begin{aligned} \min \quad & g(y) \triangleq \tfrac{1}{2}\sum_{i=1}^{m}[(|b_i| - y)_+]^2 + \gamma y \\ \text{s.t.} \quad & y \geq 0 \end{aligned} \tag{7.3.21}$$

Taking the derivative of the $g(y)$, we have

$$\nabla g(y) = \gamma - \sum_{i=1}^{m}(|b_i| - y)_+ = \gamma - \sum_{i\in I}(|b_i| - y), \text{ where } I = \{i \,|\, y \leq |b_i|\}$$

Notice that $\nabla g(y)$ is nondecreasing function for $y \geq 0$, and when $y = 0$, $\nabla g(y) = \gamma - \sum_{i=1}^{m}|b_i|$. If we denote the y^* to be the optimal solution of (7.3.21), then we have the following

$$\begin{cases} y^* = 0 & \text{if} \quad \gamma \geq \sum_{i=1}^{m}|b_i| \\ y^* = \frac{\sum_{i\in I}|b_i| - \gamma}{|I|} & \text{if} \quad \gamma < \sum_{i=1}^{m}|b_i| \end{cases}.$$

Once we obtained the optimal y^*, based on (7.3.20) we can easily get the corresponding optimal solution a^* of the subproblem as below

$$a_i^* = \begin{cases} b_i & \text{if} \quad |b_i| \leq y^* \\ y^* & \text{if} \quad |b_i| > y^* \end{cases}$$

□

We can apply the result to solving the subproblem (7.3.18) by letting $\mathbf{a} = \mathbf{a}^t$, $\mathbf{b} = (\hat{\mathbf{a}}^t)^{[k-1]} - \frac{(\mathbf{h}^t)^{[k]}}{L_i^{[k-1]}}$, and $\gamma = \rho/L_i^{[k-1]}$ in the lemma's setting. Finally, we summarize our BCD methods for tensor CP low-rank decomposition in Algorithm 19.

Algorithm 19 One step of BCD method for solving tensor low-rank decomposition

Input: N-way tensor $\mathcal{X}$. $(k-1)$-th step factor matrices $A_1^{[k-1]}, \cdots, A_N^{[k-1]}$.

1: **for** $n = 1, 2, \cdots$ **do**
2: Compute $P_n^{[k-1]}$, $L_n^{[k-1]}$, $\omega_n^{[k-1]}$, and $H_n^{[k]}$ according to (7.3.13)–(7.3.16).
3: Let $\hat{A}_n^{[k-1]} = A_n^{[k-1]} + \omega_n^{[k-1]}(A_n^{[k-1]} - A_n^{[k-2]})$.
4: Update $A_n^{[k]}$ according to (7.3.12) with the closed form in (7.3.17) or (7.3.19).
5: **end for**

Output: k-th step factor matrices $A_1^{[k]}, \cdots, A_N^{[k]}$.

7.3.3 Convergence

The convergence results of the Algorithm 19 holds as long as the objective function in (7.3.9) satisfies the so-called Kurdyka-Łojasiewicz (KL) property, which is defined as follows.

Definition 7.3.2 *A function $\psi(x)$ satisfies the Kurdyka-Łojasiewicz (KL) property at point $\bar{x} \in \mathrm{dom}(\partial\psi)$ if there exist $\phi(s) = cs^{1-\theta}$ for some $c > 0$ and $\theta \in [0, 1)$, and a certain neighborhood* U *of $\bar{x}$, such that the KL inequality holds*

$$\phi'(|\psi(x) - \psi(\bar{x})|)\mathrm{dist}(0, \partial\psi(x)) \geq 1, \quad \forall x \in \mathrm{U} \cap \mathrm{dom}(\partial\psi) \ \ \textit{and} \ \ \psi(x) \neq \psi(\bar{x}), \tag{7.3.22}$$

where $\mathrm{dom}(\partial\psi) \triangleq \{x : \partial\psi(x) \neq \varnothing\}$ *and* $\mathrm{dist}(0, \partial\psi(x)) \triangleq \min\{\|y\| : y \in \partial\psi(x)\}$.

The functions which have the KL property contain a large amount of classes of functions in applications, for instance, semi-algebraic, subanalytic and strongly convex functions all have the KL property (see the discussion in Section 2.2 of [126] for more details). Our objective function in (7.3.9) only contains three types of functions, namely, $\|\cdot\|_2^2$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$. Notice that all those three functions are semi-algebraic function, and the function in (7.3.9) are linear compositions of $A_i, i = 1, 2 \ldots, N$ or their components products, we readily know that the objective function in (7.3.9) is also semi-algebraic,

hence it has the KL property and the convergence of the algorithm can be guaranteed.

7.4 Tensor Low-Rank Completion

In this section, we show that our algorithm for low-rank approximation can be modified to solve low-rank completion problems, which is to recover low-rank tensor from partial data. It is known that many problems in signal possessing, computer vision and MRI can be formulated into the completion problems [53, 81], since sometimes part of data are missing by various reasons. There have already been some algorithms for tensor completion problems [48, 53, 81], but we did not observe a method work directly on the tensor itself. Specifically, in the previous literature, the authors first unfold the tensor into some related matrix by rearranging the positions of the elements and then consider the completion problem of the resulting matrix. For instance, Liu et al [81] and Gandy et al [48] studied the low-n-rank recovery of a tensor, which is the average of the rank of all mode matrices. However, the relationship between the n-rank and the CP rank is still unclear. Recently, Jiang et al. [70] showed that for a super-symmetric tensor, it is rank one in the sense of CP if and only if its square unfolding matrix is also rank one. To our best knowledge, the work in this chapter is the first attempt to study Tucker and CP rank completion problem. Numerical results show that our algorithms can recover missing data from only a small amount of samples.

CP Low-Rank Completion

One formulation of CP low-rank completion problem is given by

$$\min_{A_1,A_2,\cdots,A_N} \frac{1}{2}\|\mathcal{P}_\Omega(\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!])\|^2 + \rho\, \mathbf{rank}_{CP}([\![A_1, A_2, \cdots, A_N]\!]) \quad (7.4.23)$$

where $\Omega \subset [I_1] \times [I_2] \cdots \times [I_N]$ is the index set of the observed entries of $\mathcal{X}$ and $\mathcal{P}_\Omega(\mathcal{X})$ keeps the entries of $\mathcal{X}$ in Ω and sets the remaining elements to zero. Again, we take

advantage of the relationship between group sparsity and low CP rank structure, and end up with the following problem:

$$\begin{aligned} & \min_{A_1,A_2,\cdots,A_N} \tfrac{1}{2}\|\mathcal{P}_\Omega(\mathcal{X} - [\![A_1, A_2, \cdots, A_N]\!])\|^2 + \rho\left(\sum_{i=1}^{N}\sum_{t=1}^{R}\|\mathbf{a}^{i,t}\|_2\right) \\ \triangleq \;& f(A_1, A_2, \cdots, A_N) + \rho\left(\sum_{i=1}^{N} r_i(A_i)\right) \end{aligned} \tag{7.4.24}$$

Since $\mathcal{P}_\Omega$ is essentially a linear mapping, it is easy to verify that the above problem fit into the regularized multiconvex optimization framework. Indeed, Algorithm 19 can be modified to solve (7.4.24). The only difference lies in computing the gradient of function $f(\cdot)$ which involves projection on the index set Ω. To calculate the partial gradient with respect to A_n, we notice that

$$f(A_1, A_2, \cdots, A_N) = \frac{1}{2}\|\mathcal{P}_{\Omega(n)}(A_n(A_N \odot \cdots A_{n+1} \odot A_{n-1} \cdots A_1)^{\mathrm{T}} - X_{(n)})\|^2$$

where $\Omega(n)$ is the mode-n unfolding of Ω. After careful examination, we have the gradient for factor matrices:

$$\nabla_{A_n} f = \mathcal{P}_{\Omega(n)}(A_n(A_N \odot \cdots A_{n+1} \odot A_{n-1} \cdots A_1)^{\mathrm{T}} - X_{(n)})\,(A_N \odot \cdots A_{n+1} \odot A_{n-1} \cdots A_1).$$

We refer the reader [73] for more about projection index set. After applying Algorithm 19 with new gradient, we get the final solution.

Tucker Low-Rank Completion

In Tucker low-rank completion problem, we want to obtain a low Tucker rank tensor that approximate the original tensor well from partial information. Since the Tucker rank is a vector, the low Tucker rank means that the values for all components of this vector are small. Following the discussion of group sparsity and Tucker low-rank effect,

we arrive at the following formulation:

$$\begin{aligned}&\min_{\mathcal{G},A_1,A_2,\cdots,A_N} \frac{1}{2}\|\mathcal{P}_\Omega(\mathcal{X}-\mathcal{G}\times_1 A_1\times_2 A_2\cdots\times_N A_N)\|^2+\rho\left(\sum_{i=1}^{N}\sum_{t_i=1}^{R_i}\|\mathbf{a}^{i,t_i}\|_2\right)\\ \triangleq\;& f(\mathcal{G};A_1,A_2,\cdots,A_N)+\rho\left(\sum_{i=1}^{N} r_i(A_i)\right).\end{aligned} \tag{7.4.25}$$

Here the first term is the standard tensor completion formulation, and the second term is the penalty term which can lead us to the low-rank effect.

In fact, the idea and algorithm in CP low-rank decomposition and completion can be used for solving Tucker low-rank completion. For instance, if we want to apply Algorithm 19, the update for the factor matrices $A_1,\cdots,A_N$ is almost the same as the CP case, except we have different gradient here:

$$\nabla_{A_n} f=\mathcal{P}_{\Omega(n)}(A_nG_{(n)}(A_N\otimes\cdots A_{n+1}\otimes A_{n-1}\cdots A_1)^{\mathrm{T}}-X_{(n)})(A_N\otimes\cdots A_{n+1}\otimes A_{n-1}\cdots A_1)G_{(n)}^{\mathrm{T}}$$

and accordingly

$$P_n^{[k-1]}=(A_N^{[k-1]}\otimes\cdots A_{n+1}^{[k-1]}\otimes A_{n-1}^{[k]}\cdots A_1^{[k]})(G_{(n)}^{[k-1]})^{\mathrm{T}}$$

where $G_{(n)}$ is the mode-n unfolding of core tensor $\mathcal{G}$.

Different from CP situation, Tucker low-rank completion requires to update the core tensor after $A_1,\cdots,A_n$ have been updated in each iteration. One should notice that updating the core tensor $\mathcal{G}$ involves a lot of computational details. The reason is that the extrapolation way for updating factor matrices (7.3.12) will make $A_1,\cdots,A_N$ no longer orthonormal. Therefore, no closed form of $\mathcal{G}$ can be written out. In our approach, we need to matricize the core tensor into a matrix, but at this time it doesn't matter which mode we choose to unfold the core tensor. For simplicity, we use 1-mode unfolding $G_{(1)}$

to update G. Specifically,

$$f(G_{(1)}) = \frac{1}{2}\|\mathcal{P}_{\Omega(1)}(A_1 G_{(1)}(A_N \otimes \cdots A_3 \otimes A_2)^{\mathrm{T}} - X_{(1)})\|^2.$$

Then

$$\nabla_{G_{(1)}} f = A_1^{\mathrm{T}} \mathcal{P}_{\Omega(1)}(A_1 G_{(1)}(A_N \otimes \cdots A_3 \otimes A_2)^{\mathrm{T}} - X_{(1)})(A_N \otimes \cdots A_3 \otimes A_2)$$

We take $L_G^{[k-1]} = \|(A_N^{[k]})^{\mathrm{T}} A_N^{[k]}\| \times \cdots \|(A_1^{[k]})^{\mathrm{T}} A_1^{[k]}\|$. In addition, let $\hat{G}_{(1)}^{[k-1]} = G_{(1)}^{[k-1]} + \omega^{[k-1]}(G_{(1)}^{[k-1]} - G_{(1)}^{[k-2]})$, where $\omega^{[k-1]}$ is defined as (7.3.15). So

$$\hat{H}_{G_{(1)}}^{[k]} = (A_1^{[k]})^{\mathrm{T}} \mathcal{P}_{\Omega(1)}(A_1^{[k]} \hat{G}_{(1)}^{[k-1]}(A_N^{[k]} \otimes \cdots A_3^{[k]} \otimes A_2^{[k]})^{\mathrm{T}} - X_{(1)})(A_N^{[k]} \otimes \cdots A_3^{[k]} \otimes A_2^{[k]})$$

would be the gradient of $f(\hat{G}_{(1)}^{[k-1]})$. Then, we are able to derive new $G_{(1)}^{[k]}$ by

$$G_{(1)}^{[k]} = \hat{G}_{(1)}^{[k-1]} - \frac{\hat{H}_{G_{(1)}}^{[k]}}{L_G^{[k-1]}} \tag{7.4.26}$$

Finally, we transform $G_{(1)}^{[k]}$ back to the core tensor tensor $G^{[k]}$ and finish the updating.

7.5 Numerical Results

In this section, we test our algorithms for 3-way tensor low-rank decomposition and completion. All tests are performed with Tensor Toolbox of version 2.5 [6]. Our algorithms are terminated whenever the condition $\|V_k - V_{k+1}\| < tol = 0.005$ holds, where V_k is either the objective value of (7.3.6) for the decomposition problem or objective value of (7.4.24) and (7.4.25) for the completion problem in the kth iteration.

In the upcoming tables, we refer *Deviation* to the relative error from the original

tensor, $Rank/R.$ to the rank of the recovered tensor, $Iter$ to the iteration number and $Time/T$ to the running time of solving the problem. In the tables for completion problem, SR is referred to the percentage of the sampling data of the original tensor $\mathcal{X}$.

Tensor Low-Rank Decomposition

Since Tucker decomposition can be accomplished in polynomial time in the literature, we focus on our algorithm performance in CP low-rank decomposition. As for a 3-way tensor, we initially randomly generate a tensor $\mathcal{X}$ which has the dimension of $30 \times 30 \times 30$, i.e. $\mathcal{X} \in \mathbb{R}^{30\times30\times30}$. Moreover, we set the CP rank of $\mathcal{X}$ to be 4 which is reasonably low compared with the dimension of the given tensor $\mathcal{X}$. To justify our algorithm for the low-rank decomposition, we assume that we are absolutely ignorant about the initial rank of the $\mathcal{X}$ which had been set as 4. Without this knowledge, we simply begin with a randomly chosen tensor $\hat{\mathcal{X}} \in \mathbb{R}^{30\times30\times30}$, and its CP rank is 20, namely the three factor matrices $A, B, C \in \mathbb{R}^{30\times20}$. As we have proposed two different types of group sparsity regularization, namely $\ell_{1,2}$-norm and $\ell_{1,\infty}$-norm. We have applied both of them in our test, and expect that the CP rank of the final solution $\mathcal{X}^*$ that returned by our algorithm will be as small as 4. The results are provided in Table 7.1, in which we find the CP rank is very close to 4 and the deviation is also acceptably small. It shows both the $\ell_{1,2}$-norm and $\ell_{1,\infty}$-norm regularization algorithms for the decomposition can obtain a quite accurate tensor with respect to the original tensor $\mathcal{X}$ and possess the low-rank property as well. Moreover, we can find that regarding the accuracy and processing time, the $\ell_{1,2}$-norm regularization slightly outperforms the $\ell_{1,\infty}$-norm regularization. Thus we would only focus on $\ell_{1,2}$-norm regularization in the remaining tests.

We also conduct another experiment comparing our algorithm of BCD scheme with the traditional alternating least squares (ALS) algorithm. In this comparison, we first randomly generate a tensor $\mathcal{X}$ which belongs to $\mathbb{R}^{20\times20\times20}$. We set the CP rank of the original tensor $\mathcal{X}$ to be within a relatively low random level which is between 2

and 8, so we do not know the exact rank of the tensor. As the previous experiment suggests, we expect our algorithm can give us a low-rank decomposition. We also run ALS with the rank of either 4 or 20. Moreover, we can add a post-processing procedure to our algorithm, that is we run the ALS algorithm based on the rank that return by our algorithm when it terminated. We expect this procedure can further enhance the accuracy of our approximation, and denote this approach as ALSBCD in our table. The numerical results are summarized in Table 7.2. We can see that the ALS can not give us a low-rank approximation, it either fails to get low-rank or fails to get a satisfactory accuracy. However, combining ALS with our algorithm can indeed improve the accuracy.

Tensor Low-Rank Completion

We test our algorithms for 3-way tensor completion problems. The general setting is quite similar to that in low-rank decomposition. As for the CP low-rank completion, we again randomly generate an original tensor $\mathcal{X}$ which has the dimension of $30 \times 30 \times 30$ i.e. $\mathcal{X} \in \mathbb{R}^{30\times30\times30}$. We also set the CP rank of the original tensor $\mathcal{X}$ to be 4. Then we randomly obliterate some certain amount of elements of the tensor, so we get a tensor with some missing data. To run our algorithm, we simply began with a randomly chosen tensor $\tilde{\mathcal{X}} \in \mathbb{R}^{30\times30\times30}$ associated with the CP rank of 20. We hope the final result $\mathcal{X}^*$ obtained by our algorithm will be as close as possible to the original tensor $\mathcal{X}$.

For Tucker low-rank completion, we randomly generate an original tensor $\mathcal{Y}$ which has the dimension of $15 \times 15 \times 15$ i.e. $\mathcal{Y} \in \mathbb{R}^{15\times15\times15}$. The Tucker rank of this tensor is set to be $(3, 3, 3)$. Like the situation of CP completion, we assume there are some missing data in the original tensor $\mathcal{Y}$ and expect our algorithm will make the final result $\mathcal{Y}^*$ as close as possible to the original tensor $\mathcal{Y}$.

From the Table 7.3 and Table 7.4, we see that the deviation is small when $SR = 0.3$ and much smaller when $SR = 0.6$. It shows that the performance of CP and Tucker low-rank completion with our method is quite reliable. One may notice that it cost more

CPU time for Tucker completion. This is because the core tensor update requires more effort than the matrix update. Nevertheless, since our method directly deals with the Tucker rank, it provides an way to recover a Tensor with low Tucker rank.

Recently, Liu, et al [81] proposed tensor completion using the matrix nuclear norm as a convex program.

$$\min_{\mathcal{X}} \sum_{n=1}^{N} \alpha_n \|X_{(n)}\|_* \quad \text{subject to} \quad \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{M}) \tag{7.5.27}$$

where α_n are pre-determined weights satisfying $\sum_n \alpha_n = 1$. They proposed three algorithms to solve the above problem and its relaxed versions, including classic low-rank tensor completion (SiLRTC), fast low-rank tensor completion (FaLRTC), and high accuracy low-rank tensor completion (HaLRTC). Among those three, FaLRTC is the most stable and efficient algorithm. So we compare the Tucker low-rank completion with FaLRTC on a three-way tensor. Each tensor is of size $15 \times 15 \times 15$ and is generated with Tucker rank $3 \times 3 \times 3$. The sample ratios are set as 0.3 and 0.7. The performances of two methods are shown in Table 7.5. It suggests that our algorithm has higher accuracy when the sample ratio is large for the Tucker rank structure though it consumes more time. Investigating the efficiency of the implementation for Tucker low-rank completion can be one of the future directions.

Moreover, we also compare the CP low-rank completion with the Tucker low-rank completion when they are applied to the same tensor. Namely, at each instance, we generate one original tensor, and then we applied both methods for tensor completion problems to that particular tensor. The original tensor we generate is either low CP rank or low Tucker rank, and we expect the performances of those two methods are distinct. We can find the numerical results in Table 7.6, where the Type of Ori. denotes the type of the low-rank structure of the original tensor. In fact, as we expected, the result shows

that the methods are more suitable for the tensor which has the corresponding structure, i.e., when the tensor has CP low-rank structure it is more preferable to use the CP low-rank completion method. The same logic applied to the Tucker case.

Among all those numerical tests, the regularization parameter ρ appears in both decomposition and completion models. If ρ is too large, the rank of the obtained tensor will be very low, but the error of approximation would be very high, and vice versa when ρ is too small. But there is not a clear and systematic way of choosing the parameter. In fact, a proper parameter ρ would vary from case to case. As a result, for different instances, the tuning process of ρ will consist of multiple trials of different values, and for that tuning process, one may choose the bisection method to perform the trials.

No.	$\ell_{1,\infty}$				$\ell_{1,2}$			
	Deviation	Rank	$Iter_{\ell_{1,\infty}}$	$Time_{\ell_{1,\infty}}$	Deviation	Rank	$Iter_{\ell_{1,2}}$	$Time_{\ell_{1,2}}$
1	2.690e-003	4	3013	56.77	9.764e-004	5	1878	16.12
2	3.038e-003	4	1901	45.52	9.230e-004	4	2690	26.26
3	3.894e-003	4	2034	48.55	1.222e-003	4	1019	11.33
4	4.473e-003	4	2270	54.52	1.366e-003	4	996	12.10
5	4.160e-003	4	2118	52.07	1.303e-003	4	1289	12.11
6	3.916e-003	4	2863	65.63	1.416e-003	4	1240	12.14
7	3.824e-003	4	2329	68.12	1.214e-003	4	1115	9.20
8	3.038e-003	4	1901	48.62	9.230e-004	4	2690	31.14
9	2.745e-003	4	2793	67.45	8.617e-004	4	1278	14.53
10	2.690e-003	4	3013	110.11	9.764e-004	5	1878	17.84

Table 7.1: CP low-rank decomposition through $\ell_{1,\infty}$ and $\ell_{1,2}$-norm regularization. Original tensor in dimension $30 \times 30 \times 30$, with CP rank 4.

No.	BCD			
	Deviation	R.	Iter	Time
1	9.68e-005	7	10963	147.56
2	8.80e-005	7	9034	124.03
3	1.08e-004	8	11064	145.89
4	9.69e-005	7	8040	109.42
5	8.43e-005	8	8203	111.75
1	8.36e-005	5	11526	116.70
2	7.79e-005	5	10656	108.14
3	1.06e-004	5	11735	117.70
4	1.59e-004	5	8740	87.96
5	8.26e-005	5	11125	110.74

No	ALS				ALSBCD			
	Deviation	R.	Iter	Time	Deviation	R.	Iter	Time
1	5.77e-001	4	11	0.56	4.32e-006	7	580	6.89
2	5.77e-001	4	12	0.21	3.67e-006	7	31	0.47
3	5.59e-001	4	15	0.21	1.55e-005	8	12	0.25
4	5.50e-001	4	15	0.19	1.75e-005	7	14	0.26
5	5.27e-001	4	18	0.22	5.69e-006	8	14	0.28
1	2.52e-005	20	7	0.27	1.43e-005	5	16	0.16
2	2.87e-005	20	9	0.29	6.99e-006	5	16	0.15
3	2.06e-005	20	9	0.39	8.30e-006	5	255	1.81
4	4.91e-005	20	9	0.34	8.05e-006	5	14	0.14
5	6.69e-005	20	14	0.41	1.06e-005	5	541	4.42

Table 7.2: Compare different methods for CP low-rank decomposition. Randomly generate tensor in dimension $20 \times 20 \times 20$ with CP rank between 2 and 8.

No.	Deviation	Rank	Iter	Time
	SR=30%(70% missing data)			
1	1.538e-004	5	13820	91.00
2	1.575e-004	4	9858	65.38
3	2.622e-004	4	9667	62.02
4	1.410e-004	6	14700	92.64
5	4.180e-004	4	11161	75.46
	SR=60%(40% missing data)			
1	7.056e-005	5	16649	111.08
2	7.608e-005	4	11159	74.27
3	1.621e-004	6	10529	70.15
4	5.812e-005	6	17917	115.55
5	1.827e-004	4	11153	73.95

Table 7.3: CP low-rank Completion. Original tensor in dimension $30 \times 30 \times 30$, with CP rank 4.SR= Sample Ratio.

No.	Deviation	Rank	Iter	Time
	SR=30%(70% missing data)			
1	1.324e-001	(6,5,6)	8135	177.13
2	2.895e-002	(6,5,5)	16307	354.24
3	3.103e-002	(3,3,4)	19462	397.20
4	3.892e-002	(4,4,6)	12495	266.51
5	2.371e-002	(4,6,4)	20234	434.07
	SR=60%(40% missing data)			
1	1.715e-002	(3,4,4)	24316	489.06
2	1.572e-002	(7,5,4)	16732	315.63
3	9.815e-003	(4,3,5)	14372	324.10
4	1.524e-002	(4,3,3)	22248	486.58
5	2.505e-002	(5,5,3)	14877	395.43

Table 7.4: Tucker low-rank Completion. Original tensor in dimension $15 \times 15 \times 15$, with Tucker rank $(3, 3, 3)$. SR= Sample Ratio.

No.	Time	Deviation	Time	Deviation
	SR=30%(70% missing data)			
	Tucker Completion		FaLRTC	
1	395.04	1.217e-002	0.46	1.211e-002
2	414.51	1.535e-002	0.34	1.304e-002
3	370.89	1.205e-002	0.37	9.061e-003
4	563.35	1.049e-002	0.37	1.054e-002
5	451.35	9.879e-003	0.35	9.236e-003
	SR=70%(30% missing data)			
	Tucker Completion		FaLRTC	
1	448.82	4.558e-002	0.61	9.347e-002
2	236.81	5.278e-002	0.63	1.421e-001
3	466.32	6.797e-002	0.60	3.873e-001
4	375.61	6.041e-002	0.70	2.996e-001
5	310.64	5.724e-002	0.61	3.073e-001

Table 7.5: Compare performance between Tucker low-rank Completion and FaLRTC. Original tensor in dimension $15 \times 15 \times 15$, with Tucker rank $(3, 3, 3)$. SR= Sample Ratio.

Type of Ori.	No.	Deviation	Rank	Iter	Time	Deviation	Rank	Iter	Time
	SR=30%(70% missing data)								
		CP Completion				Tucker Completion			
CP low-rank structure	1	1.194e-003	4	2122	4.55	2.377e-001	(6,7,6)	12156	323.38
	2	2.139e-003	4	3469	7.26	2.864e-001	(4,5,6)	19206	513.46
	3	1.895e-003	4	3087	6.45	3.100e-001	(7,5,5)	11192	307.2
Tucker low-rank structure	1	1.193e-001	6	1757	3.96	3.038e-002	(9,8,9)	13751	367.07
	2	2.997e-001	5	250	0.57	6.053e-002	(6,9,8)	13927	372.72
	3	7.678e-002	7	4878	10.42	2.473e-002	(7,7,5)	16527	442.68
	SR=60%(40% missing data)								
CP low-rank structure	1	6.393e-004	4	2989	6.33	1.860e-001	(7,9,6)	10074	262.64
	2	1.083e-003	5	2043	4.38	2.247e-001	(5,5,3)	11803	338.47
	3	8.944e-004	4	2989	6.54	3.056e-001	(5,3,4)	11227	303.99
Tucker low-rank structure	1	4.197e-002	7	654	1.48	1.391e-002	(6,8,6)	17823	476.39
	2	3.208e-002	7	1881	4.10	7.917e-003	(6,7,5)	20768	556.28
	3	6.877e-002	6	1006	2.18	9.922e-003	(9,6,8)	13953	443.61

Table 7.6: Compare CP and Tucker low-rank completion under different structure.

Chapter 8

Conclusion

This thesis investigates scalable methods for optimization problems in machine learning. In particular, we consider the problem with structure in the form of

$$\min_{x\in\mathcal{D}}\{F(x) = f(x) + r(x)\}.$$

Many machine learning problems can be covered in this framework, consisting of one part penalizing the violation of training data (in loss function f) and one part penalizing violation of the structural assumptions, or a convex relaxation of the structural assumption (in $\mathcal{D}$ and r). Based on different needs and applications, a wide class of regularization strategies have been applied, including sparsity with ℓ_1 norm, group sparsity with $\ell_{1,2}$ norm, low-rank with nuclear norm, and some others, as introduced in Section 1.2.

With the growth of data size and model complexity, scalable methods and their properties are in huge demand to push the performance limits. This thesis conducts research on a wide class of scalable methods for structural optimization problems in machine learning, such as proximal gradient method, accelerate proximal gradient method, alternating method of multipliers, coordinate descent, and Frank-Wolfe method. A review

of standard iterations and state of art results is given in Section 1.3. In the following, we summarize the contribution of this thesis.

In Chapter 2, we present structural machine learning models and their properties that are of great interest throughout this thesis. Chapter 3 & 4 focus on a sparse model known as the LASSO model [112, 113]. We show the local linear convergence bound through a numerical linear algebra approach. This approach can be applied to a wide class of methods including proximal gradient, accelerate proximal gradient, alternating direction method of multipliers and coordinate descent.

- In Chapter 3, we show the local linear convergence of proximal gradient and accelerated proximal gradient, applied to the model LASSO problem. Since both algorithms on LASSO problem use iterative shrinkage operator, they are also known as ISTA and FISTA. Using the numerical linear algebra techniques, we show both algorithms can be modeled as matrix recurrence forms and thus the associated spectra can be used to analyze their convergence behaviors. It is shown that the method normally passes through several regimes of four types and eventually settles on a "linear regime" in which the iterates converge linearly with the rate depending on the absolute value of the second largest eigenvalue of the matrix recurrence.

  In addition, we provide a way to analyze every type of the regime. Such analysis in terms of regimes allows one to study the aspect of acceleration of FISTA. It is well known that FISTA is faster than ISTA according the worst case complexity bound. Our analysis gives another way to show how both methods behave during the whole iterations. It turns out that FISTA is not always faster than ISTA in linear convergence regime, depending on the continually growing stepsize. But in general FISTA is faster because of its acceleration in constant regime. A heuristic algorithm is developed based on this observation and exhibits very good numerical performance.

- In Chapter 4, we show the local linear convergence of ADMM and Coordinate Descent on the LASSO model. Using the same technique as in Chapter 3, we show ADMM can also be modeled as matrix recurrence and thus the spectral analysis can be applied to analyze its convergence behavior. We also establish a connection between Coordinate Descent method and Gauss-Seidel method, in a way to compare with ISTA in Chapter 3. We present numerical examples on all scalable methods in Chapter 3 & 4 for the LASSO model, to justify the theory and properties we have established.

Chapter 5 & 6 investigate the group sparse model [114, 115]. We first focus on a matrix model known as Sparse Inverse Covariance Estimation. We develop a novel estimator incorporating the group structural constraint and Frank-Wolfe method for solving it. Then we extend the estimator to a more general one with a scalable solver and show their broad applications.

- In Chapter 5, we present an inverse covariance estimator that can regularize for overlapping group sparsity, and provide better estimates when number of dimensions is much larger than number of samples. We show that the group structure itself can be used to incorporate decomposition and speed up the computation. Simulation results justify our estimators by better accuracy compared to the state-of-art methods. We further applied this model on congressman voting history data and Yahoo! finance data.

- In Chapter 6, we extend to a general structure model that can both exploit the prior knowledge and reduce computation cost. The generalization is from three aspects. First, each group norm is extended from ℓ_2 norm to any valid norm. Second, each group can be further restricted to conic constraint. Last but not least, we consider any convex and differential function as our objective. We show that the estimator

in Chapter 5 is a special case, and bring into more examples as applications.

Chapter 7 moves on to tensor, a natural extension from vector and matrix. We explore the low-rank structure in tensor and build the structural constraint for optimization [49].

- In Chapter 7, we put forward a method of finding tensor low-rank decomposition without knowing the rank. We construct the connection between group sparsity and tensor rank, so that different group structural regularizations are proposed. The resulting problem is a regularized multi-convex problem and can be solved efficiently through block coordinate update. The numerical performances have shown that this approach is effective, especially when the given tensor has a low-rank structure. We also apply it on the tensor completion problem as a major application.

In summary, the thesis presents novel research on scalable optimization methods for machine learning. We show a wide class of structures for different models. We explore many convergence properties of scalable methods that can take advantage of the structure. We apply the models and methods to the real world problems.

References

[1] J. Agler, W. Helton, S. McCullough, and L. Rodman. Positive semidefinite matrices with a given sparsity pattern. *Linear algebra and its applications*, 107:101–149, 1988.

[2] M. S. Andersen, J. Dahl, and L. Vandenberghe. Logarithmic barriers for sparse matrix cones. *Optimization Methods and Software*, 28(3):396–423, 2013.

[3] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*, 2015.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, Dec 2008.

[5] A. Argyriou, R. Foygel, and N. Srebro. Sparse prediction with the k-support norm. In *Advances in Neural Information Processing Systems*, pages 1457–1465, 2012.

[6] B. W. Bader and T. G. Kolda. Matlab tensor toolbox version 2.5. 2012.

[7] O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *J. Mach. Learn. Res.*, 9:485–516, June 2008.

[8] O. Banerjee, L. El Ghaoui, A. d'Aspremont, and G. Natsoulis. Convex optimization techniques for fitting sparse gaussian graphical models. In *Proc. of the 23rd ICML*, pages 89–96. ACM, 2006.

[9] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2(1):183–202, 2009.

[10] J. Bigot, R. J. Biscay, J.-M. Loubes, and L. Muñiz-Alvarez. Group LASSO estimation of high-dimensional covariance matrices. *Journal of Machine Learning Research*, 12(Nov):3187–3225, 2011.

[11] D. Boley. Local linear convergence of the alternating direction method of multipliers on quadratic or linear programs. *SIAM Journal on Optimization*, 23(4):2183–2207, 2013.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[13] K. Bredies and D. Lorenz. Linear convergence of iterative soft-thresholding. *Journal of Fourier Analysis and Applications*, 14(5-6):813–837, 2008.

[14] J. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, March 2010.

[15] E. Candès and T. Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *Ann. Statist.*, 35(6):2313–2351, 12 2007.

[16] E. J. Candès and Y. Plan. Near-ideal model selection by ℓ_1 minimization. *The Annals of Statistics*, 37:2145–2177, 2009.

[17] E. J. Candès and Y. Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Trans. Inf. Theor.*, 57(4):2342–2359, April 2011.

[18] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009.

[19] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[20] E. J. Candès, T. Strohmer, and V. Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *CoRR*, abs/1109.4499, 2011.

[21] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Trans. Inf. Theor.*, 51(12):4203–4215, December 2005.

[22] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition. *Psychometrika*, 35:283–319, 1970.

[23] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky. The convex geometry of linear inverse problems. *Foundations of Computational mathematics*, 12(6):805–849, 2012.

[24] C. Chen, B. He, Y. Ye, and X. Yuan. The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. *Mathematical Programming*, 155(1):57–79, Jan 2016.

[25] C. Chen, B. He, and X. Yuan. Matrix completion via an alternating direction method. *IMA Journal of Numerical Analysis*, 32(1):227–245, 2012.

[26] S. Chen, D. L. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1998.

[27] P. L. Combettes and J. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[28] J. Dahl, L. Vandenberghe, and V. Roychowdhury. Covariance selection for non-chordal graphs via chordal embedding. *Optimization Methods & Software*, 23(4):501–520, 2008.

[29] A. d'Aspremont, O. Banerjee, and L. El Ghaoui. First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and Applications*, 30(1):56–66, 2008.

[30] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math.*, 57(11):1413–1457, 2004.

[31] V. DeMiguel, L. Garlappi, and R. Uppal. Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *Review of Financial Studies*, 22(5):1915–1953, 2009.

[32] A. P. Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.

[33] W. Deng and W. Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *UCLA CAM technical report*, 2012.

[34] D. L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theor.*, 52(4):1289–1306, April 2006.

[35] J. C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

[36] M. Friedlander E. van den Berg, M. Schmidt and K. Murphy. Group sparsity via linear-time projection. *Technical Report, Department of Computer Science, University of British Columbia*, 2008.

[37] J. Eckstein and D. P. Bertsekas. An alternating direction method for linear programming. *Technical report, MIT*, 1990.

[38] J. Eckstein and D. P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55:293–318, 1992.

[39] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[40] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer Publishing Company, Incorporated, 1st edition, 2010.

[41] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista. Fast consensus by the alternating direction multipliers method. *IEEE Transactions on Signal Processing*, 59(11):5523–5537, Nov 2011.

[42] J. Mairal F. Bach, R. Jenatton and G. Obozinski. Structured sparsity through convex optimization. *Statist. Sci.*, 27, 2012.

[43] M. Fortin and R. Glowinski. Augmented lagrangian methods: applications to the numerical solution of boundary-value problems. *North-Holland Pub. Co.*, 1983.

[44] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[45] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical LASSO. *Biostatistics*, 9(3):432–441, 2008.

[46] J. Friedman, T. Hastie, and R. Tibshirani. Applications of the LASSO and grouped LASSO to the estimation of sparse graphical models. Technical report, Technical report, Stanford University, 2010.

[47] L. Fuchs. Recovery of exact sparse representations in the presence of bounded noise. *IEEE Trans. on I.T*, pages 3601–3608, 2005.

[48] S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(6), 2011.

[49] X. Gao, B. Jiang, and S. Tao. Recovering low CP/Tucker ranked tensors, with applications in tensor completion. *Pacific Journal of Optimization*, 2015.

[50] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 69–77, 2011.

[51] G. H. Gene and V. Loan. *Matrix Computations (4th Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 2013.

[52] R. Glowinski and P. Le Tallec. Augmented lagrangian and operator-splitting methods in nonlinear mechanics. *SIAM*, 1989.

[53] D. Goldfarb and Z. Qin. Robust low-rank tensor recovery: Models and algorithms. *SIAM J. Matrix Anal. Appl.*, 35(1):225–253, 2014.

[54] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[55] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th Annual International Conference on Machine Learning*, ICML '10, pages 399–406, 2010.

[56] A. Griewank and P. L. Toint. On the existence of convex decompositions of partially separable functions. *Mathematical Programming*, 28(1):25–49, 1984.

[57] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial hermitian matrices. *Linear algebra and its applications*, 58:109–124, 1984.

[58] J. Hastad. Tensor rank is np-complete. *J. Algorithms*, 11, 1990.

[59] B. He and X. Yuan. On the o(1/n) convergence rate of douglas-rachford alternating direction method. *SIAM J. Numer. Anal.*, 50:700–709, 2012.

[60] B. He and X. Yuan. Local linear convergence of the alternating direction method of multipliers for quadratic programs. *SIAM J. Numer. Anal.*, 2013.

[61] N. J. Higham. Computing the nearest correlation matrix - a problem from finance. *IMA Journal of Numerical Analysis*, 2002.

[62] M. Hong and Z. Luo. On the linear convergence of the alternating direction method of multipliers. *arXiv:1208.3922*, 2013.

[63] M. J. Hosseini and S. Lee. Learning sparse gaussian graphical models with overlapping blocks. In *Advances in Neural Information Processing Systems*, pages 3801–3809, 2016.

[64] K. Hou, Z. Zhou, A. So, and Z. Luo. On the linear convergence of the proximal gradient method for trace norm regularization. In *Advances in Neural Information Processing Systems 26*, pages 710–718. 2013.

[65] A. S. Householder. *The Theory of Matrices in Numerical Analysis*. A Blaisdell book in the pure and applied sciences. 1964.

[66] C. Hsieh, M. Sustik, I. Dhillon, P. Ravikumar, and R. Poldrack. BIG & QUIC: Sparse inverse covariance estimation for a million variables. In *Advances in Neural Information Processing Systems*, pages 3165–3173, 2013.

[67] J. Huang and T. Zhang. The benefit of group sparsity. *The Annals of Statistics*, 2010.

[68] M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *ICML*, pages 427–435, 2013.

[69] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 457–464. ACM, 2009.

[70] B. Jiang, S. Ma, and S. Zhang. Tensor principal component analysis via convex optimization. *Math. Program.*, 2014.

[71] J. Jiang, H. Wu, Y. Li, and R. Yu. Three-way data resolution by alternating slice-wise diagonalization (asd) method. *J. Chemometrics*, 14, 2000.

[72] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Frontiers in Applied Mathematics. SIAM, 1995.

[73] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3), 2009.

[74] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[75] J. B. Kruskal. Rank, decomposition and uniqueness for 3-way and n-way arrays. *Mult. Data Anal.*, 1989.

[76] S. Lacoste-Julien and M. Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems 28*, pages 496–504. 2015.

[77] J. Lam, C.and Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Ann. Statist.*, 37(6B):4254–4278, 12 2009.

[78] J. Liang, J. Fadili, and G. Peyré. Local linear convergence of forward–backward under partial smoothness. In *Advances in Neural Information Processing Systems 27*, pages 1970–1978. 2014.

[79] T. Lin, S. Ma, and S. Zhang. On the global linear convergence of the admm with multiblock variables. *SIAM Journal on Optimization*, 25(3):1478–1497, 2015.

[80] P. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979.

[81] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25:208–220, 2013.

[82] J. W. H. Liu. The multifrontal method for sparse matrix solution: Theory and practice. *SIAM review*, 34(1):82–109, 1992.

[83] H. Markowitz. Portfolio selection. *The journal of finance*, 7(1):77–91, 1952.

[84] R. Mazumder and T. Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *JMLR*, 13(Mar):781–794, 2012.

[85] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the LASSO. *The annals of statistics*, pages 1436–1462, 2006.

[86] Jean-Jacques Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math*, 255:2897–2899, 1962.

[87] S. Negahban and M. J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 823–830, 2010.

[88] S. Negahban, B. Yu, M. J. Wainwright, and P. K. Ravikumar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009.

[89] Y. Nesterov. A method of solving a convex programming problem with convergence rate $o(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[90] G. Obozinski, L. Jacob, and J. Vert. Group LASSO with overlaps: the latent group LASSO approach. *arXiv preprint arXiv:1110.0413*, 2011.

[91] B. O'Donoghue and E. J. Candès. Adaptive restart for accelerated gradient scheme. *Found. Comp. Math.*, 15(3):715–732, 2013.

[92] J. M. Ortega. *Numerical Analysis : A Second Course*. Society for Industrial and Applied Mathematics, 1990.

[93] M. R. Osborne, B. Presnell, and B. A. Turlach. On the LASSO and its dual. *Journal of Computational and Graphical Statistics*, 9:319–337, 1999.

[94] P. Moulin P. Johnstone. A Lyapunov analysis of FISTA with local linear convergence for sparse optimization. *arXiv:1502.02281*, 2015.

[95] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[96] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electron. J. Statist.*, 5:935–980, 2011.

[97] R. Rebonato and P. Jäckel. The most general methodology to create a valid correlation matrix for risk management and option pricing purposes.

[98] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, August 2010.

[99] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[100] R. Rockafellar. On the maximal monotonicity of subdifferential mappings. *Pacific J. Math.*, 33, 1970.

[101] R. Rockafellar. On the maximality of sums of nonlinear monotone operators. *Transactions of the American Mathematical Society*, 149:75–88, 1970.

[102] R. Rockafellar. Augmented lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1:97–116, 1976.

[103] R. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14, 1976.

[104] A. Rohde and A. B. Tsybakov. Estimation of high-dimensional low-rank matrices. *Ann. Statist.*, 39(2):887–930, 04 2011.

[105] B. Rolfs, B. Rajaratnam, D. Guillot, I. Wong, and A. Maleki. Iterative thresholding algorithm for sparse inverse covariance estimation. In *Advances in Neural Information Processing Systems*, pages 1574–1582, 2012.

[106] A. J. Rothman, P. J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electron. J. Statist.*, 2:494–515, 2008.

[107] A. Saha and A. Tewari. On the finite time convergence of cyclic coordinate descent methods. *CoRR*, 2010.

[108] K. Scheinberg, S. Ma, and D. Goldfarb. Sparse inverse covariance selection via alternating linearization methods. In *Advances in neural information processing systems*, pages 2101–2109, 2010.

[109] K. Scheinberg and I. Rish. Sinco-a greedy coordinate ascent method for sparse inverse covariance selection problem. *preprint*, 2009.

[110] K. Scheinberg and I. Rish. Learning sparse gaussian markov networks using a greedy coordinate ascent approach. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 196–212. Springer, 2010.

[111] Y. Sun and L. Vandenberghe. Decomposition methods for sparse matrix nearness problems. *SIAM Journal on Matrix Analysis and Applications*, 36(4), 2015.

[112] S. Tao, D. Boley, and S. Zhang. Convergence of common proximal methods for ℓ_1 regularized least squares. In *International Joint Conference on Artificial Intelligence*. 2015.

[113] S. Tao, D. Boley, and S. Zhang. Local linear convergence of ISTA and FISTA on the LASSO problem. *SIAM Journal on Optimization*, 2016.

[114] S. Tao, Y. Sun, and D. Boley. Submatrix constrained inverse covariance estimation. *NIPS Workshop on Learning in High Dimension with Structure*, 2016.

[115] S. Tao, Y. Sun, and D. Boley. Inverse covariance estimation with group structure. *International Joint Conference on Artificial Intelligence*, 2017.

[116] H. Taylor, S. Bank, and J. McCoy. Deconvolution with the ℓ_1 norm. *Geophysics*, 44:39–52, 1979.

[117] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[118] R. J. Tibshirani. The LASSO problem and uniqueness. *Electronic Journal of Statistics*, 7(0):1456–1490, 2013.

[119] K. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 2010.

[120] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *SIAM Journal on Optimization*, 2008.

[121] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1-2):387–423, 2009.

[122] L. Vandenberghe, M. S. Andersen, et al. *Chordal graphs and semidefinite optimization*. now publishers Incorporated, 2015.

[123] S. A. Vavasis. *Nonlinear Optimization: Complexity Issues*. Oxford University Press, Inc., New York, NY, USA, 1991.

[124] D. P. Wipf and B. D. Rao. An empirical bayesian strategy for solving the simultaneous sparse approximation problem. *IEEE Trans. Sig. Proc*, pages 3704–3716, 2007.

[125] S. Wright. Coordinate descent algorithms. *Math. Program.*, 151(1):3–34, 2015.

[126] Y. Xu and W. Yin. A block coordinate descent method for multi-convex optimization with applications to nonnegative tensor factorization and completion. *SIAM J. Imaging Sci.*, 27, 2013.

[127] J. Yang and Y. Zhang. Alternating direction algorithms for l1-problems in compressive sensing. *SIAM J. Sci. Comput.*, 33(1):250–278, February 2011.

[128] L. Yuan, J. Liu, and J. Ye. Efficient methods for overlapping group LASSO. In *Advances in Neural Information Processing Systems*, pages 352–360, 2011.

[129] M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

[130] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 68, 2009.

[131] X. Yuan. Alternating direction method for covariance selection models. *Journal of Scientific Computing*, 51(2):261–273, 2012.

[132] H. Zhang, W. Yin, and L. Cheng. Necessary and sufficient conditions of solution uniqueness in ℓ_1 minimization. *CoRR*, abs/1209.0652, 2012.